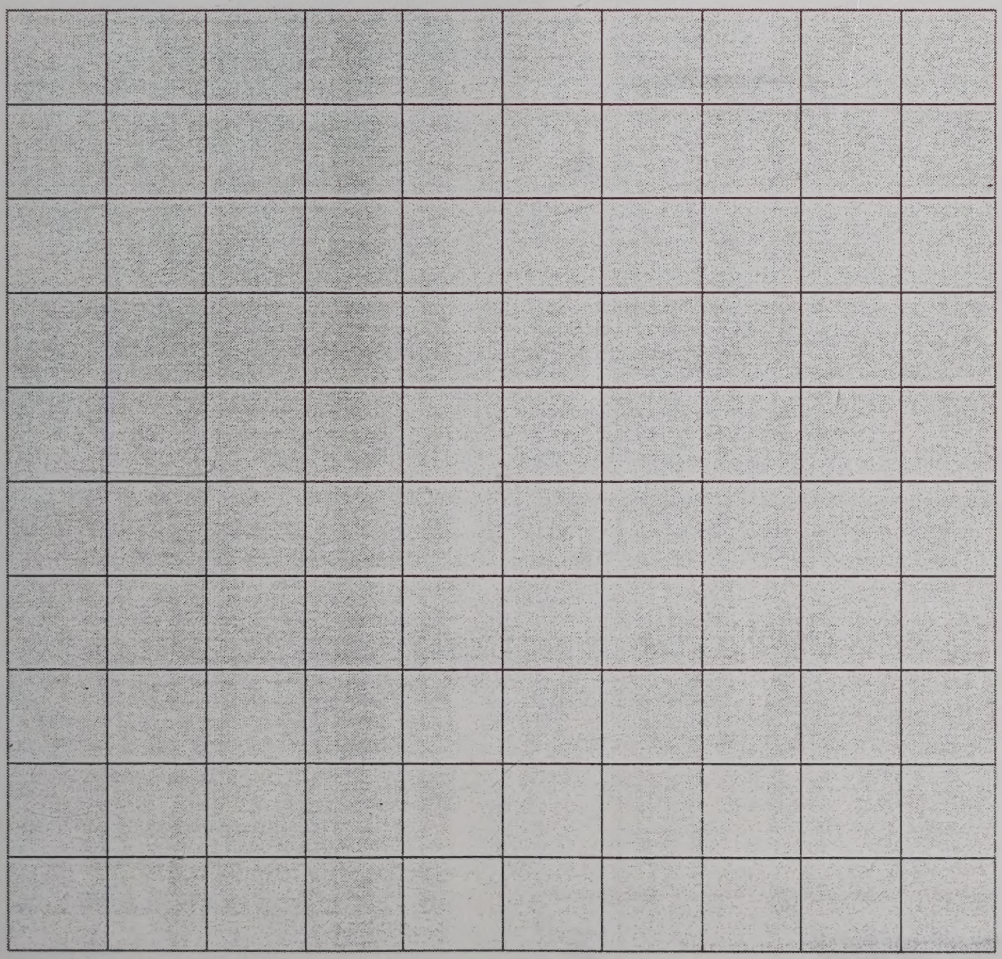
SIEMENS

Protocol Tester K1197

User Manual



© Siemens AG 1996

Published by
Siemens AG
Automatisierungstechnik

Copying of this document and giving it to others and the use or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights are reserved in the event of the grant of a patent or the registration of a utility model or design.

© Siemens AG 1995 All rights reserved

Subject to technical alteration

Siemens Aktiengesellschaft

C73000–X6076–C601–3

Order No.: C73000–G6076–C601–3
Available from AUT33 P AZ Berlin
Printed in the Federal Republic of Germany
AG 0696 En

Contents

Protocol Tester K1197
Dual Interface
Conformance Tests
RTE for TTCN.MP

1	RTE C73000–M6076–C600
2	SIETEM C73000–B6076–C511
3	
4	
5	

Are you ...

interested in learning more about this product or another area of communications test equipment? Write to:

Siemens AG
AUT 33 V
Kommunikationsmeßtechnik
Wernerwerkdamm 5
D–13623 Berlin

Fax (+49 30) 386 27934

Do you ...

wish to inform us on your ideas to this manual? Please send your suggestions indicating the order No. of this manual:

Siemens AG
AUT 33 E
Kommunikationsmeßtechnik
Redaktion
Wernerwerkdamm 5
D–13623 Berlin

Fax (+49 30) 386 27934

If there is a CE symbol on the device, then:

CE

The product described in this manual complies with the statutory regulations on electromagnetic compatibility of devices (EMC) or the regulations of the guideline of the council from 03.05.1989 for adaptation of the statutory regulations of the member countries on electromagnetic compatibility of devices (89/336/EEC).

RUN Time Environment for TTCN Compiler

User Manual for Protocol Tester K1197

SIEMENS AG

Software Version	:	V2.0
System Version	:	MF 3.0
Revision Date	:	March, 1996
Engineering	:	AUT 33 E1 / LENZ
Quality control	:	MWB KQP
Document No.	:	C73000–M6076–C600–3

© Siemens AG 1995

Published by
Siemens AG
Automatisierungstechnik

Copying of this document and giving it to others and the use or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights are reserved in the event of the grant of a patent or the registration of a utility model or design.

Subject to technical changes

Siemens Aktiengesellschaft

Order No.: C73000-M6076-C600-3
Available from AUT33 P AZ Berlin
Printed in the Federal Republic of Gerrany
AG 0396 En

Contents

		Page
1	**Introduction**	**1–1**
1.1	General	1–1
1.2	Getting Started	1–1
1.2.1	Installation	1–1
1.3	RTE Loading Instructions for ISDN Base	1–2
1.3.1	RTE Load Menu	1–3
1.3.2	TTCN Test Suite Menu	1–4
2	**Menu System**	**2–1**
2.1	Parameter and Test Selection Menu	2–1
2.2	LOG_SELECT	2–2
2.3	RTE Parameter Menu	2–9
3	**RTE Usage**	**3–1**
3.1	General	3–1
3.2	Directing Output	3–1
3.3	Control of Data Objects	3–1
3.4	Displaying of Declarations and Constraints	3–2
3.5	Sending of Constraints	3–2
3.6	Invoking of Test Steps/Cases/Groups/Suites	3–2
3.7	Resetting of Test Suite and Test Case Variables	3–3
3.8	Test Suite Structure Menu	3–3
3.9	Test Suite Parameter Menu	3–4
4	**Output Descriptions**	**4–1**
4.1	Constraints Output	4–1
4.2	Debug Output from RUN Time	4–1
5	**Sample Test Session**	**5–1**
5.1	Start the Session for ISDN Base	5–1

6 Other Words 6–1

7 Runtime Error Messages 7–1

8 Reference List 8–1

1 Introduction

1.1 General

This manual describes the usage of the Run Time Environment (RTE) for Executable Test Suites (ETSs) generated by the ISO DIS/IS 9646–3 TTCN.MP compiler software for K1197 Protocol Tester.

The TTCN.MP compiler generates ETSs that can be stored as binary files. The RTE is intended to run the ETS files on a K1197 protocol tester.

Details about general parameters, test case selections and test case run may be obtained from SIETEM© manual [4]. SIETEM functionalities concerning the Indentation Compiler are not supported in this RTE software.

This application program needs a RAM extension! **Please refer to [2] for more information. For installation refer also to chapter 2 of [3]. The RAM extension may be installed in any vacant slot as shown in figure 7.6–1.**

1.2 Getting Started

1.2.1 Installation

According to the configuration of your K1197 protocol tester copy the appropriate RTE program from floppy disks to your hard disk. Make sure that you don't overwrite equally named files!

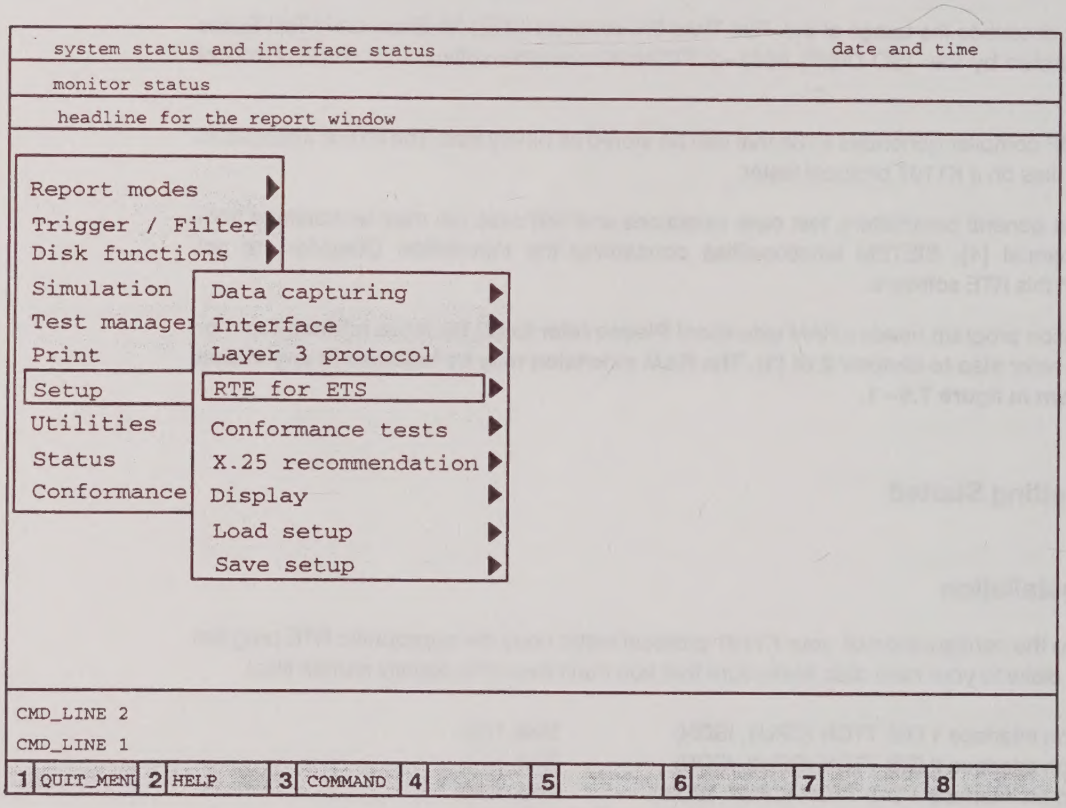
RTE for s0 on interface 1 DIS TTCN (CPU1, ISDN)	Disk 1/10
RTE for s0 on interface 2 DIS TTCN (CPU3, ISDN)	Disk 1/10
RTE for s2m on interface 1 DIS TTCN (CPU7, ISDN)	Disk 1/10
RTE for s2m on interface 2 DIS TTCN (CPU8, ISDN)	Disk 1/10
RTE for s0 on interface 1 IS TTCN (CPU1, ISDN)	Disk 1/10
RTE for s0 on interface 2 IS TTCN (CPU3, ISDN)	Disk 1/10
RTE for s2m on interface 1 IS TTCN (CPU7, ISDN)	Disk 1/10
RTE for s2m on interface 2 IS TTCN (CPU8, ISDN)	Disk 1/10

Note: The RTE has to use the same CPU for an ETS where it was compiled on.

SIETEM© (SIEMENS AG) Siemens TTCN Test Manager

1.3 RTE Loading Instructions for ISDN Base

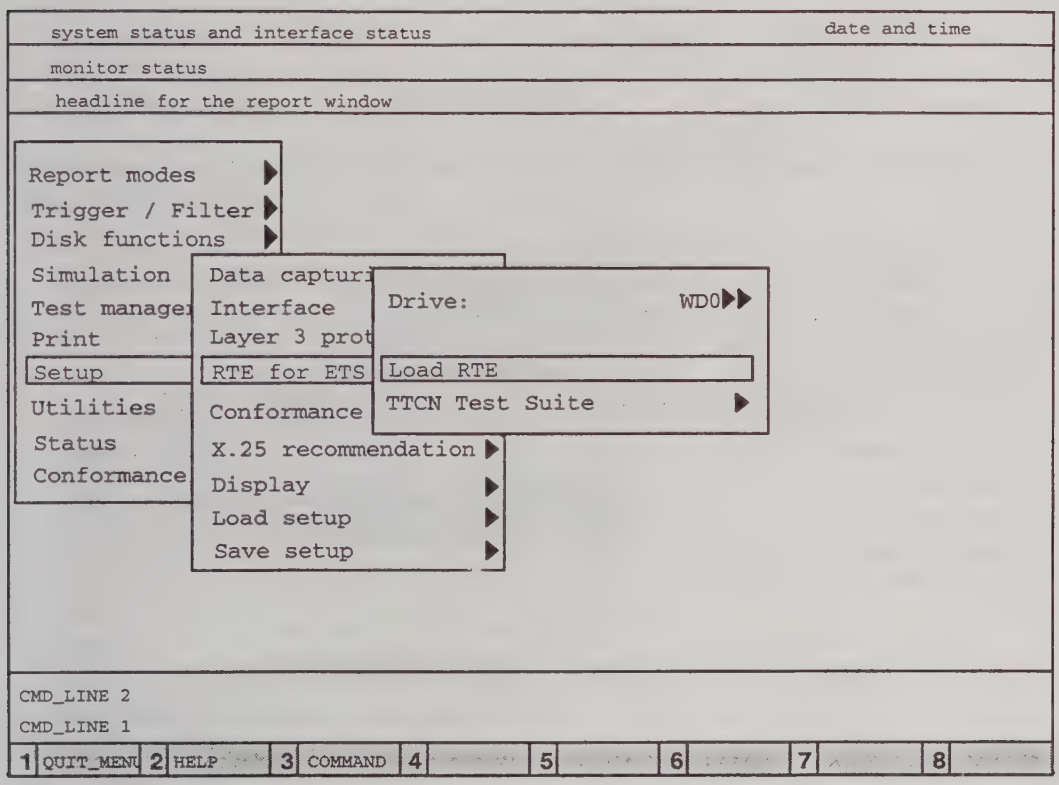
Please load the modified application program from the RTE package and then the national variant from the application package via menu system as described in application program manual[1]. The RTE as well as the executable test suite can also be loaded via menu system. The "Cursor right" and " Cursor left" keys control the movement through the menu levels.



MENU SELECTION:	
RTE for ETS	: Select RTE submenus

Note: The setup menu is found when pressing function key F1 from the application.

1.3.1 RTE Load Menu

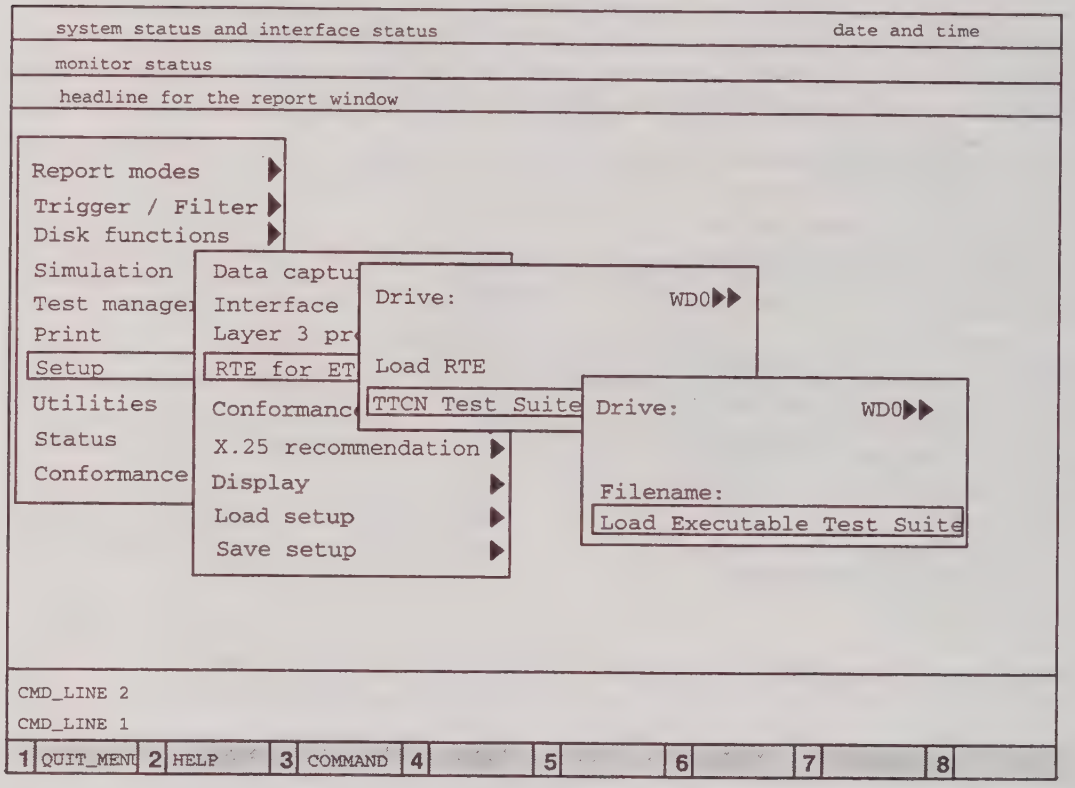


MENU SELECTION:	
Load RTE	: Load Run Time Environment
TTCN Test Suite	: Select TTCN test suite submenu

Note: When the RTE menu does not show up, it may be that any other ISDN application program has been loaded. Please load the application program from the RTE diskette or from that partition of your harddisk the RTE software has been copied to.

Note: Only one RTE may be loaded. It is not possible to load several RTEs at the same time on different interfaces.

1.3.2 TTCN Test Suite Menu



MENU SELECTION:	
Filename	: Insert the filename of the ETS binary
Load Executable Test Suite	: Load Executable Test Suite

Note: Before loading the RTE, the intended national variant shall be loaded first. To load an ETS binary, at first the intended national variant and then the RTE shall be loaded. The RTE is not included in the general SETUP mechanism.

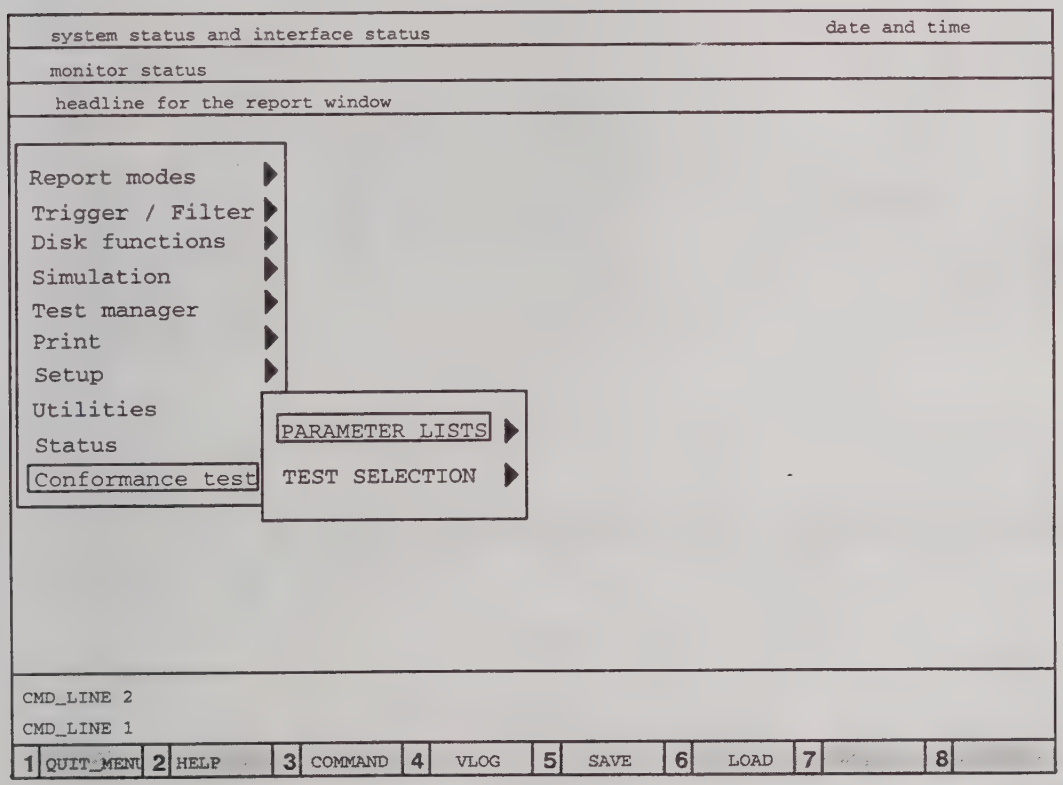
Caution: The hash compiler QUICKY should be always turned OFF on load and save operations (type `OFF QUICKY`).

Caution: After usage of the RTE other application programs should not be started without prior reboot.

2 Menu System

2.1 Parameter and Test Selection Menu

After loading the ETS, this menu is available. Parameter setting as well as test case selection and execution is menu driven.

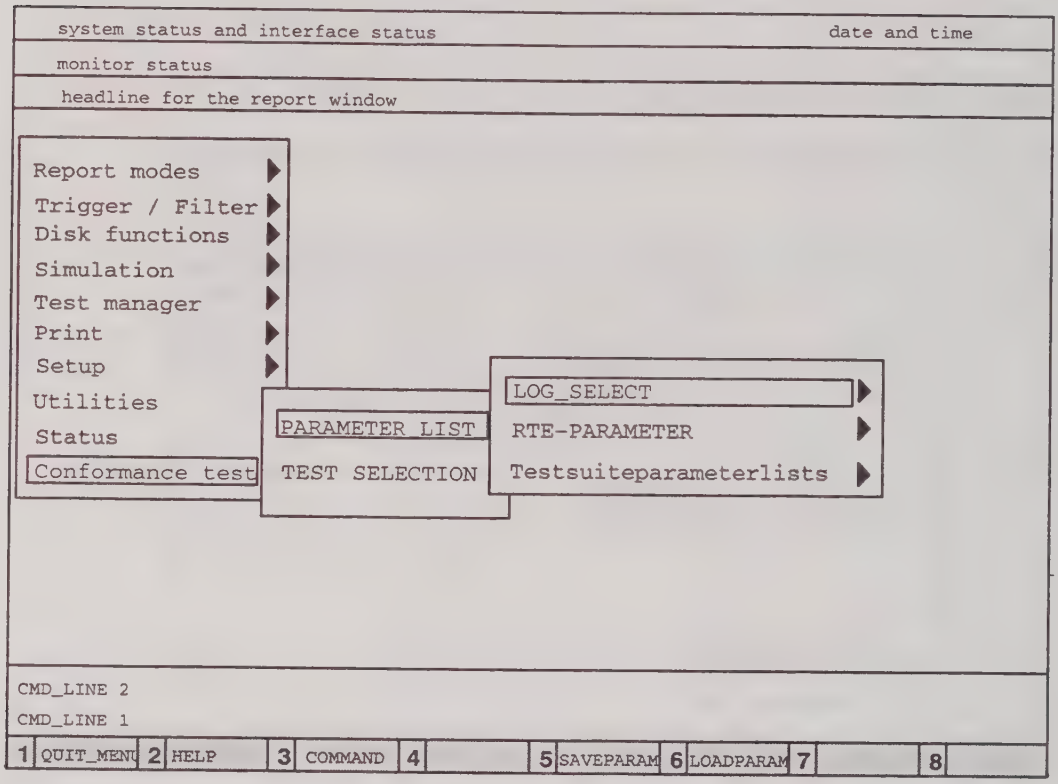


MENU SELECTION:

PARAMETER LIST	: List of the PICS and PIXIT variables
TEST SELECTION	: Test Cases tree to be selected and executed
FUNCTION KEYS:	
QUIT_MENU	: Close the menu window
HELP	: Open a help window
COMMAND	: Open FORTH–Interpreter window
VLOG	: Show a list of Verdict of Test Cases wich are finished
SAVE	: Save the actual values of all PICS/PIXIT parameters
LOAD	: Load previously saved PICS/PIXIT parameters

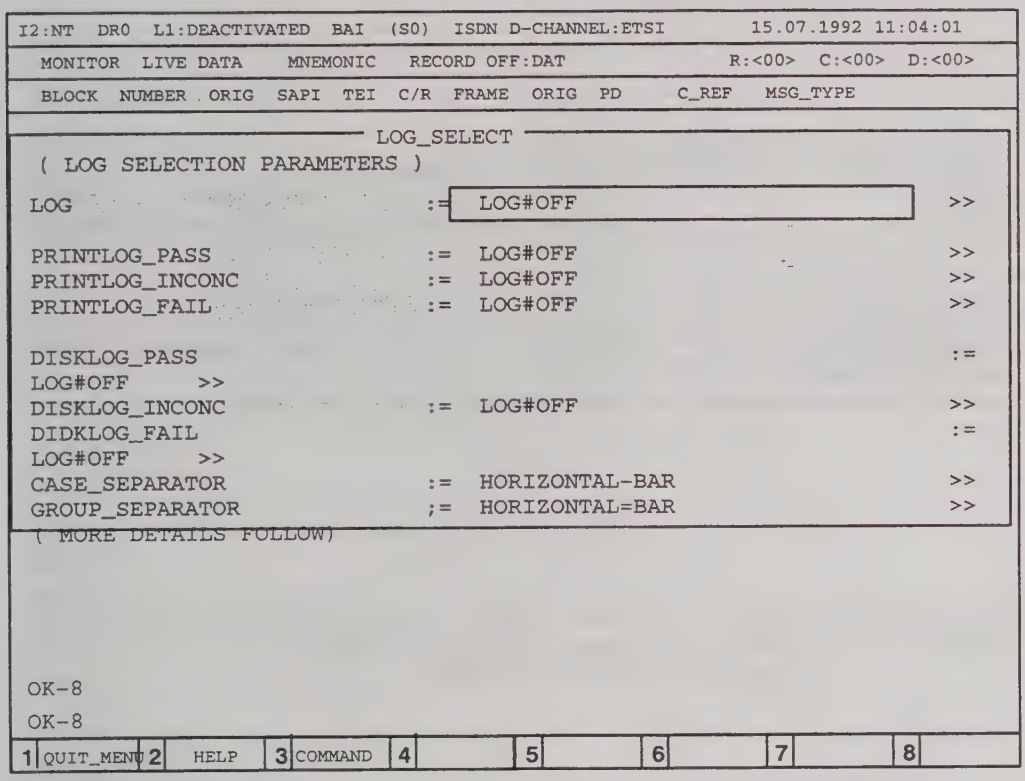
Note: Do not use key *DEL* to delete characters of the name of the PICS/PIXIT list! Use key *BS* instead.

2.2 LOG_SELECT

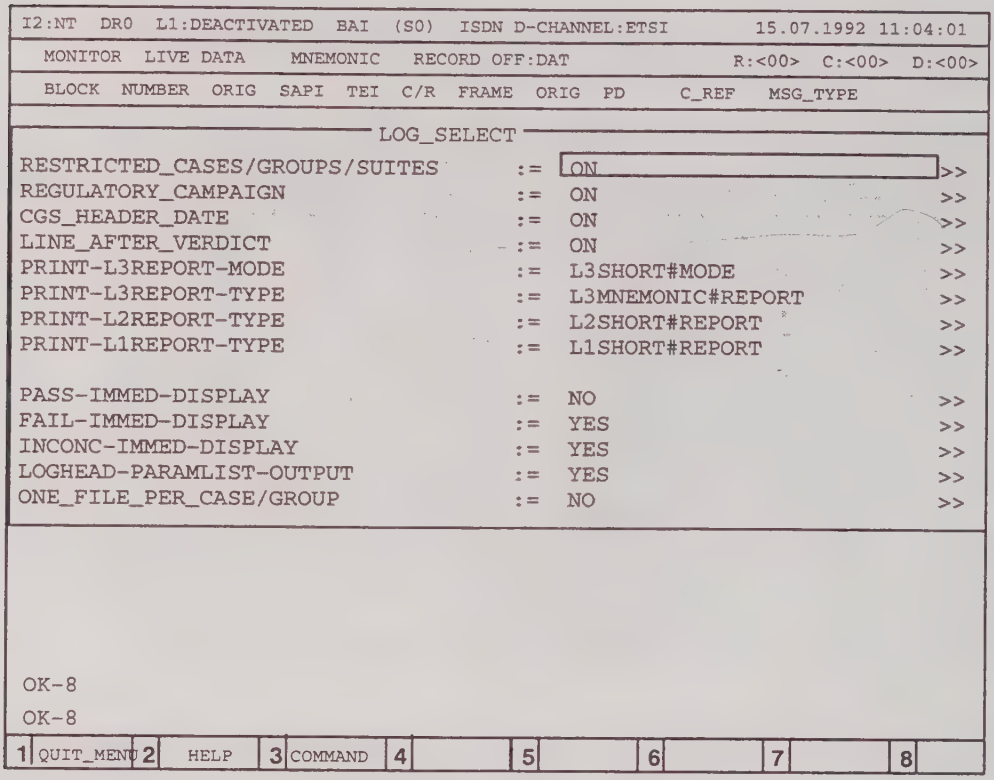


MENU SELECTION:	
LOG_SELECT	: Conformance log parameter selection
RTE–PARAMETER	: RTE parameter selection
FUNCTION KEYS:	
SAVEPARAM	: Save single parameter list
LOADPARAM	: Load single parameter list

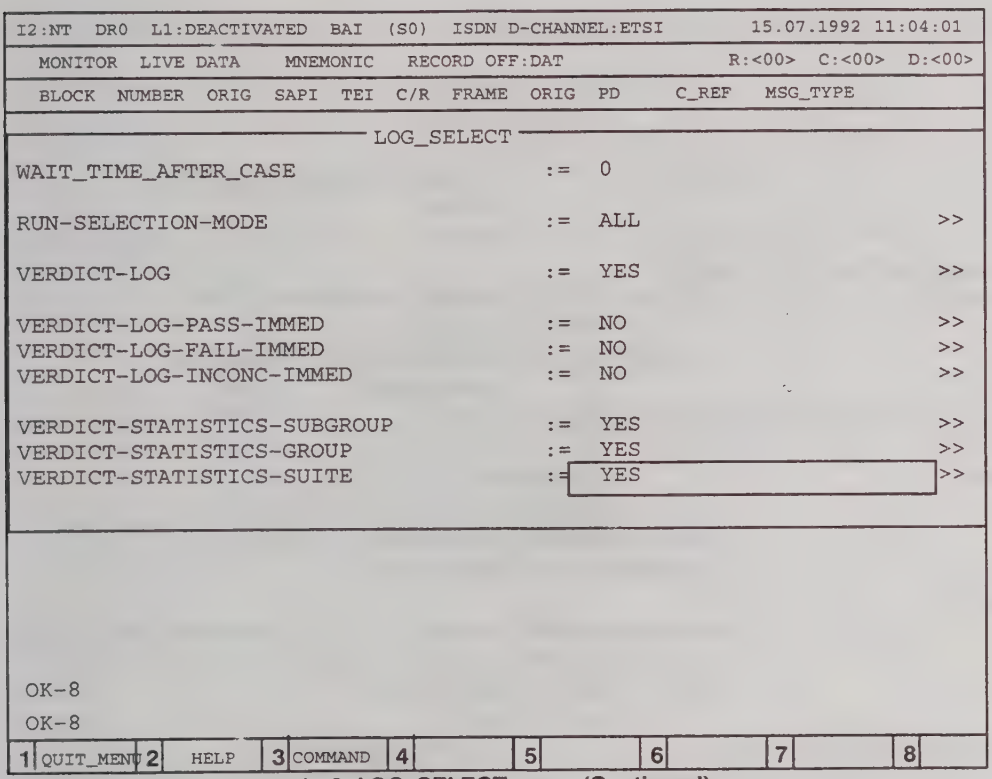
1–1 to 1–3 show the possible parameter selections of the LOG_SELECT parameter group, which control type and extension of the test reports. The parameters can be reached with the Cursor keys and can be changed by opening another window with "Cursor Right" key.


I2:NT DR0 L1:DEACTIVATED BAI (S0) ISDN D-CHANNEL:ETSI 15.07.1992 11:04:01
MONITOR LIVE DATA MNEMONIC RECORD OFF:DAT R:<00> C:<00> D:<00>
BLOCK NUMBER ORIG SAPI TEI C/R FRAME ORIG PD C_REF MSG_TYPE
LOG_SELECT
(LOG SELECTION PARAMETERS)
LOG := LOG#OFF >>
PRINTLOG_PASS := LOG#OFF >>
PRINTLOG_INCONC := LOG#OFF >>
PRINTLOG_FAIL := LOG#OFF >>
DISKLOG_PASS :=
LOG#OFF >>
DISKLOG_INCONC := LOG#OFF >>
DIDKLOG_FAIL :=
LOG#OFF >>
CASE_SEPARATOR := HORIZONTAL-BAR >>
GROUP_SEPARATOR ;= HORIZONTAL=BAR >>
(MORE DETAILS FOLLOW)
OK-8
OK-8
1 QUIT_MENU 2 HELP 3 COMMAND 4 5 6 7 8

1–1: LOG_SELECT menu


I2:NT DR0 L1:DEACTIVATED BAI (S0) ISDN D-CHANNEL:ETSI 15.07.1992 11:04:01
MONITOR LIVE DATA MNEMONIC RECORD OFF:DAT R:<00> C:<00> D:<00>
BLOCK NUMBER ORIG SAPI TEI C/R FRAME ORIG PD C_REF MSG_TYPE
LOG_SELECT
RESTRICTED_CASES/GROUPS/SUITES := ON >>
REGULATORY_CAMPAIGN := ON >>
CGS_HEADER_DATE := ON >>
LINE_AFTER_VERDICT := ON >>
PRINT-L3REPORT-MODE := L3SHORT#MODE >>
PRINT-L3REPORT-TYPE := L3MNEMONIC#REPORT >>
PRINT-L2REPORT-TYPE := L2SHORT#REPORT >>
PRINT-L1REPORT-TYPE := L1SHORT#REPORT >>
PASS-IMMED-DISPLAY := NO >>
FAIL-IMMED-DISPLAY := YES >>
INCONC-IMMED-DISPLAY := YES >>
LOGHEAD-PARAMLIST-OUTPUT := YES >>
ONE_FILE_PER_CASE/GROUP := NO >>
OK-8
OK-8
1 QUIT_MENU 2 HELP 3 COMMAND 4 5 6 7 8

1–2: LOG_SELECT menu (Continued)


I2:NT DR0 L1:DEACTIVATED BAI (S0) ISDN D-CHANNEL:ETSI 15.07.1992 11:04:01
MONITOR LIVE DATA MNEMONIC RECORD OFF:DAT R:<00> C:<00> D:<00>
BLOCK NUMBER ORIG SAPI TEI C/R FRAME ORIG PD C_REF MSG_TYPE
LOG_SELECT
WAIT_TIME_AFTER_CASE := 0
RUN-SELECTION-MODE := ALL >>
VERDICT-LOG := YES >>
VERDICT-LOG-PASS-IMMED := NO >>
VERDICT-LOG-FAIL-IMMED := NO >>
VERDICT-LOG-INCONC-IMMED := NO >>
VERDICT-STATISTICS-SUBGROUP := YES >>
VERDICT-STATISTICS-GROUP := YES >>
VERDICT-STATISTICS-SUITE := YES >>
OK-8
OK-8
1 QUIT_MENU 2 HELP 3 COMMAND 4 5 6 7 8

1–3: LOG_SELECT menu (Continued)

The following table gives an overview over the meaning of the different parameters.

Parameter	possible values	Meaning
LOG	LOG#OFF	
	LOG#CASE	The Log will be written after each test case.
PRINTLOG_PASS/INCONC/FAIL (depends on test results)	LOG#OFF	
	LOG#SHORT	Only test name and verdict will be printed.
	LOG#LONG	Test name, traces and the Verdict will be printed.

The same agreement is valuable for the parameter DISKLOG.

CASE_SEPARATOR	OFF	
	HORIZONTAL-BAR	----------
	HORIZONTAL=BAR	==========
	FORM#FEED	for the printer
GROUP_SEPARATOR	OFF	
	HORIZONTAL-BAR	----------
	HORIZONTAL=BAR	==========
	FORM#FEED	for the printer
RESTRICTED_CASES/GROUPS/SUITES	ON	Only the selected Test-cases/groups/suites will be started.
	OFF	All test cases will be run, also the not selected.

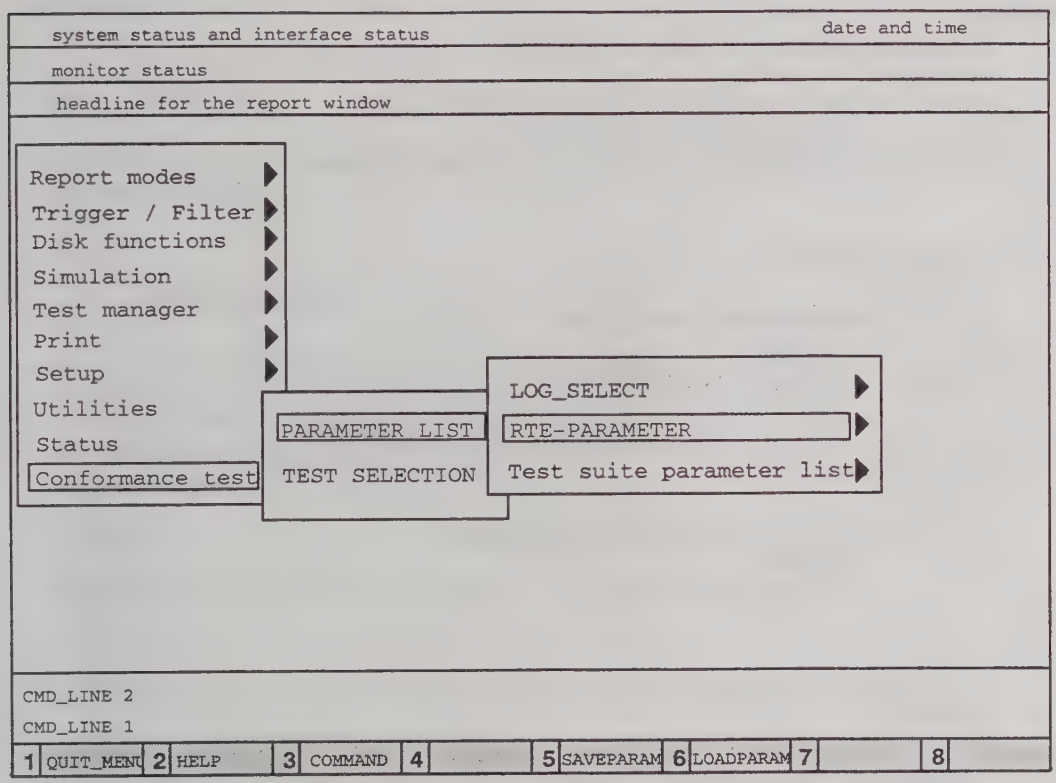
Remark: Selected test cases will be marked with "+" in the selection menu.

REGULATORY_CAMPAIGN	ON	Only Test cases which make sense with the actual PICS-selection are selected or can be selected.
	OFF	All test cases can be selected.
CGS_HEADER_DATE	ON	For each test case/group/suite the actual date will be printed into the log. If there is enough space, the actual time will be printed also.
	OFF	No time/date will be printed
LINE_AFTER_VERDICT	ON	A line will be added after each verdict.
	OFF	No line after verdict.

PRINT–L3REPORT–MODE	L3SHORT#MODE	Printing format of layer 3 Test reports.
	L3MEDIUM1#MODE	
	L3MEDIUM2#MODE	
	L3COMPLETE#MODE	
PRINT–L3REPORT–TYPE	L3HEX#REPORT	More details about layer 3 printing format.
	L3CHAR#REPORT	
	L3NONE#REPORT	
	L3MNEMONIC#REPORT	
PRINT–L2REPORT–TYPE	L2HEX#REPORT	Printing format of layer 2 test reports.
	L2CHAR#REPORT	
	L2SHORT#REPORT	
	L2NONE#REPORT	
	L2COMPLETE#REPORT	
PRINT–L1REPORT–TYPE	L1SHORT#REPORT	Printing format of layer 1 test reports
	L1NONE#REPORT	
	L1COMPLETE#REPORT	
PASS/FAIL/INCONC–IMMED–DISPLAY	YES	Temporary Verdicts will immediately be shown in brackets.
	NO	Only one verdict will be printed at the end of the test case.
LOGHEAD–PARAMLIST–OUTPUT	YES	The complete actual status of the PICS/PIXIT will be shown at the beginning of the test log.
	NO	No status information.
ONE_FILE_PER_CASE/GROUP	YES	The log of each test case will be saved in a different file.
	NO	The whole test suit will be saved in one file.
WAIT_TIME_AFTER_CASE	ZAHL	Timeout after a test case in 1/10 seconds.
RUN–SELECTION–MODE	ALL	All selected Tests can be executed.
ONLY CASES WITH		Only Tests with Verdict no equal to can be executed.

VERDICT <> PASS	PASS	
VERDICT–LOG	YES	Enables the verdict recording.
	NO	Disables the verdict recording.
VERDICT–LOG		
PASS/FAIL/INCONC–IMMED	YES	Enables the additional step verdict.
	NO	Disables the additional step verdict.
VERDICT–STATISTICS–SUBGROUP –GROUP –SUITE	YES	Enables the verdict–statistic.
	NO	Disables the verdict–statistic.

2.3 RTE Parameter Menu



MENU SELECTION:	
LOG_SELECT	: Conformance log parameter selection
RTE–PARAMETER	: RTE parameter selection
FUNCTION KEYS:	
SAVEPARAM	: Save single parameter list
LOADPARAM	: Load single parameter list

The following parameters are contained in the RTE parameter list:

WATCH–PDU–NAME

Possible Values: ALL or any valid PDU identifier
Meaning:
Name of the PDU, for that CODEBUG should produce output.
ALL: Output about all receive constraints.
Identifier: Name of the PDU for that output should be generated.

EVDEBUG

Possible Values: Integer
Meaning:
Controls output of received events in trace.
0: No output.
<>0: Output about received events.

TIMDEBUG

Possible Values: 0 or 1
Meaning:
Controls output of timer events in trace (e.g, timeouts).
0: No output.
<>0: Output about timer events.

CODEBUG

Possible Values: 0,1 or 2
Meaning:
Controls format of output of receive constraint matching results in trace. See chapter Debug Output from Run Time in this manual for more informations.
0: No output.
1: Verbal output about receive constraints to the PDU named by WATCH–PDU–NAME.
2: Detailed output about receive constraints to the PDU named by WATCH–PDU–NAME.

3 RTE Usage

3.1 General

The RTE is intended to be used to run ETSs generated by the TTCN.MP ISO DIS/IS 9646–3 Compiler on ISDN base.

3.2 Directing Output

The output may be directed to screen (default), to trace (screen and capture) and to other channels in debug mode or when manually testing variables, constraints or PDUs.The following does not apply, when running test cases, steps or suites. In this case use the test log commands in the parameter list `LOG_SELECT`. The following commands can be used to direct output during debug operations:

`V.SCREEN`

directs all subsequent output to the screen.

`V.PRINT`

directs all subsequent output to the printer. (Printer configuration must be set correctly).

`V.DISK`

directs all subsequent output to the current disk device, output is stored in 64 kB editor files with names `VDISKxx.V` where xx is the number of the file starting with 00 (the number of files is restricted to 100). The variable `VDISK-NO` contains the number of the next file to be written. The last file has to be written manually with the command `VDISKWTD`. A buffer on RAM extension at address 0X3F0000 is bought for this routines.

`V.REMOTE`

allows to send output to the remote port.

3.3 Control of Data Objects

Since all TTCN Declarations and Constraints are compiled as FORTH structures, they may be invoked by typing their Ids. The RTE is able to distinguish several "methods" to know what is to be done with the object invoked (e.g. sending, receiving, executing or displaying). These methods are set automatically by the RTE over the method stack MSTACK. For test purposes the user may set the method for display. The usage is described below. At all times after setting and using a method, the method should be finally removed from method stack by the command `MMETHOD>`. The variable `MMETHOD` contains the actual method.

3.4 Displaying of Declarations and Constraints

The command sequence:

```
MDEBUG   >MMETHOD   AnyId   MMETHOD>
```

displays the contents of any TTCN Variable,Constant, PDU, Constraint etc. with name `AnyId`. This output is suited for TTCN objects with more than value contents like types, timers or constraints. The value of any data object like variables, constants or PDUs may be displayed using the command

```
V.DATAOBJ AnyId*.
```

Note: Before applying the above command set the output direction e.g. V.SCREEN for screen.

3.5 Sending of Constraints

The command sequence:

```
!MP   PCOId   PDUId   ConsId   SENDC   MMETHOD>
```

sends or displays the Constraint `ConsId` (depending on the parameter `CONSTRAINT.TEST`). `!MP` sets the method `MTX` for sending and prepares the send buffer. `PDUId` is the name of the PDU for the constraint. `PCOId` is the name of the PCO where the message has to be sent. `SENDC` performs the final sending.

Note: If the constraint contains parameters, the parameters should be set before sending the constraint. The appropriate assignment word should be used (e.g. `PAR_ASSIGN` for literal values).

3.6 Invoking of Test Steps/Cases/Groups/Suites

The most convenient way is to use the menu system CONFORMANCE TESTS/ TEST SELECTION. The command sequence:

```
0   TM=   Case/Group/SuiteId
```

invokes the appropriate Test Case/Group/Suite.

The command sequence:

```
0   TM-STEP=   StepId
```

invokes the appropriate Test Step.

Note: Invoking a test step alone (outside a test case) will not automatically reset test case variables. If resetting of variables is desired, this has to be done manually (see below) before the test step is invoked. Multiple invoking of test steps outside test cases may cause an `MSTACK` overflow error, since the `MSTACK` is not reset in this case. Therefore the `MSTACK`

* Only RTEs for IS Compiler Versions

should be reset manually before invoking a test step using the command `RESET_MSTACK`. **When a test step is left, it automatically searches for the test case from which it was invoked. If an invoking test case cannot be found, an error message is given but this does not take any influence in the test verdict.**

3.7 Resetting of Test Suite and Test Case Variables

The command sequence:

```
RESET_TSVAR
RESET_TCVAR
RESET_TSCON*
```

resets all Test Suite, Test Case Variables and Test Suite Constants* to their initial value.

3.8 Test Suite Structure Menu

The test cases, groups or the entire suite are accessible via the menu system. The menu TEST SELECTION opens the access to the test suite. Via function keys the execution (F8) or selection (F5, F6, F7) of test cases, groups or the entire suite are possible.

The text at the beginning of each test suite/test group/test case is marked blue on the screen followed by a detailed description of the started test part.
At the end of each test case the same presentation appears, followed by one of the possible verdicts PASS, FAIL or INCONC.

There is the following colour assignment:

PASS	marked blue
FAIL	marked red
INCONC	marked yellow

At the end of a test group or a test suite a global verdict is added in the same way. In the test run, temporary verdicts appear in brackets with the same colour assignment. For more details see SIETEM manual [4].

* Only RTEs for IS Compiler Versions

3.9 Test Suite Parameter Menu

Before running a test suite or case the TTCN parameters have to be set to correct values. The parameters of the test suite and the trace control are accessible via the `PARAMETER LISTS` menu. It is possible to store or load parameter lists via function keys. To save all parameter lists open the menu `Conformance tests` and set the cursor to the menu item `PARAMETER LISTS`. Press the function key F5 and then input a filename. To load all the parameter lists the function key F6 has to be used. **Then the PARAMETER LISTS menu has to be opened once!**
The parameter values can also be set with the following command sequence per parameter:

RTEs for DIS Compiler Versions:

```
n   PAR_ASSIGN   value   '   TS_PARid   CFA   DYN_PAR!
```

where n is the exact length in bits and value is the TTCN.MP value representation depending on the type of the parameter (e.g. the value 64 is 64 for INTEGER but '01000000'B for BITSTRING or '40'H for HEXSTRING). Empty values should be specified by an empty string e.g ''B or ''O.

RTEs for IS Compiler Versions:

```
PAR_ASSIGN   value   TS_ParId
```

where value is the TTCN.MP value representation depending on the type of the parameter (e.g. the value 64 is 64 for INTEGER but '01000000'B for BITSTRING or '40'H for HEXSTRING). Empty values should be specified by an empty string e.g ''B or ''O.

4 Output Descriptions

4.1 Constraints Output

The structure of the constraints output using the method MDEBUG is as follows:

1. name of the constraint
2. related PDU/ASP/Structured Type
3. name of the ancestor constraint (NONE if it is a base constraint)
4. list of PDU field/ ASP parameter/ Structured Type element templates
5. list of constraint value declarations (one for each PDU field/ASP parameter/Structured type element template if the constraint is a base constraint)

4.2 Debug Output from RUN Time

The structure of the receive constraint debug output during run time of the ETS is as follows (for every Field/Parameter/Element):

Fields with non wildcard value:

1. received value
2. constraint field value
3. field name and result of comparison

Fields with wildcard value

1. value stored in field
2. field name

End of constraint

1. constraint matching result (if all fields matched and the constraint did not match, the length of the received data was not right)

The values with length < 8 bit are traced as BITSTRING and longer values are traced as OCTETSTRING*. See also WATCH–PDU–NAME and CODEBUG.

* Only RTEs for IS Compiler Versions

5 Sample Test Session

5.1 Start the Session for ISDN Base

This chapter describes how to start and run through a test session for a RTE on a ISDN base software.

1st step: Preparation of the IUT

2nd step: Preparation of the K1197

Load and setup the ISDN application program (`SIGNALLING APPLICATION/ISDN SIMULATION`), the national variant (`Setup/Layer 3 protocol/LOAD VARIANT`), the RTE (`Setup/RTE for ETS/Load RTE`), the executable test suite (`Setup/RTE for ETS/TTCN Test Suite/Load Executable Test Suite`), the test suite parameters (`Conformance tests/PARAMETER LISTS/Function Key F6`) and the testmanager script (`TEST MANAGER/LOAD`), if it is not saved as binary.

3rd step: Connection between K1197 and IUT

Connect K1197 and IUT according to the relevant configuration.

4th step: Preparation of PICS/PIXIT parameters

Setup all PICS/PIXIT parameters with values according to the IUT via menu system. Set the RTE parameters for run time according your wishes, e.g. set TIMDEBUG to 1 if timer events should be traced or set CODEBUG to 1 if constraint matching results should be traced.

5th step: - Starting a Test Case / Group / Suite

Start a test case / group / suite using the `TEST SELECTION MENU` (function key F8: `RUN_SEL`) after setting the cursor to the wanted suite, group or case (To look into a suite or group use the '->' cursor key.).

6 Other Words

>MMETHOD **(method --)**
push method on method stack. That means this method will be the current method for all objects. Available methods are:
MDEBUG: all objects will generate debug output if called
MDATA: all objects will put there data pointer on stack if called (some objects put the pointer to the data pointer on the stack)
MLENGTH: all objects will put there attribute pointer on stack if called

MMETHOD> **(--)**
pop method from method stack.

RESET_MSTACK **(--)**
reset method stack.

7 Runtime Error Messages

Error:

– " PUSH_MSTACK: OVERFLOW"

Reason/Explanation:

The method stack is overflown. Please type `MMETHOD>` to clear the last entry or `RESET_MSTACK` to clear the entire stack.

Error:

– " POP_MSTACK: EMPTY"

Reason/Explanation:

The method stack is empty. Use nnn >MMETHOD to set a method.

Error:

– " TIMER DURATION NOT OK: 0.1 sec is taken as min. Duration"

Reason/Explanation:

There may be a test suite parameter for a timer duration not set.

Error:

– " No space in sendstring for pdu field ..."

Reason/Explanation:

The length of a PDU is restricted to 1024 byte.

Error:

– Test case ends with TC_ERROR and verdict INCONCLUSIVE.

Reason/Explanation:

A parameter may be set wrong or not loaded. Check the setting of the parameters.

Error:

– The RTE is searching for a TMTM.? file after starting the first test case or an incompatible version message appears.

Reason/Explanation:

The test manager script is not loaded or a wrong binary version has been loaded.

8 Reference List

[1] ISDN Emulation/Simulation, USER MANUAL FOR DUAL INTERFACE PROTOCOL TESTER K1197, Siemens AG, C73000–M6076–C175

[2] K1195 Protocol Tester, 2 Mbyte RAM Extension, ordering number S44033–K1195–R11

[3] K1197 Dual Interface Basic MANUAL, Siemens AG, C73000–G6076–C111

[4] SIETEM TTCN TEST MANAGER, USER MANUAL FOR PROTOCOL TESTER K1197, Siemens AG, C73000–B6076–C511

INDEX

Symbols

>MMETHOD, 6–1

C

COMMAND, 2–1
Constraint Test, 3–2

D

Data Objects, 3–1
DYN_PAR!, 3–4

E

ETS, 1–1

H

HEADER–STRINGS, 2–10
HELP, 2–1

I

Installation, 1–1
Invoking Cases, 3–2

L

LOAD, 2–1
Load Executable Test Suite, 1–4
LOAD MENU, 1–3
Load RTE, 1–3
LOADPARAM, 2–2, 2–9

M

MDATA, 6–1
MDEBUG, 6–1
Methoden Stack, 3–3

SIETEM®
TTCN Test Manager

User Manual for Protocol Tester K1197

SIEMENS AG

Software Version	:	V 6.0
System Version	:	MF 3.0
Revision Date	:	April, 1995
Engineering	:	AUT 33 E2 / SM
Quality Control	:	MWB KQP 2.
Document No.	:	C73000–B6076–C511–4

© Siemens AG 1994

Published by
Siemens AG
Automatisierungstechnik

Copying of this document and giving it to others and the use or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights are reserved in the event of the grant of a patent or the registration of a utility model or design.

Subject to technical changes

Siemens Aktiengesellschaft

Order No.: C73000-B6076-C511-4
Available from AUT3 K AZ Berlin
Printed in the Federal Republic of Germany
AG 0694 En

Contents

		Page
1	**INTRODUCTION**	**1–1**
1.1	SIETEM integration in K1197	1–2
1.2	Advantages of the 'SIETEM' implementation	1–3
1.3	TEST SUITE structure	1–4
1.4	SIETEM in comparison with standard notation	1–5
2	**INSTALLATION**	**2–1**
2.1	LOADING OF SIETEM SIETEM BINARY	2–1
2.1.1	ADAPTION OF THE STORAGE FOR CONFORMANCE TESTS UNDER SS NO.7	2–2
3	**MENU SYSTEM**	**3–1**
3.1	Parameter and test selection menu	3–1
3.1.1	Parameter menu	3–3
3.1.2	Test case selection menu	3–4
3.2	Special functions	3–5
3.2.1	VERDICT LOG (VLOG)	3–5
3.2.2	Statistics	3–6
4	**SIETEM parameters**	**4–1**
4.1	LOG_SELECT	4–3
4.1.1	TEST ENVIROMENT	4–3
4.1.2	LOG REPORT PARAMETERS	4–3
4.1.3	DISK LOG parameters	4–4
4.1.4	PRINT LOG parameters	4–5
4.1.4.1	REPORT PARAMETER FOR X.25	4–5
4.1.4.2	Report parameters for ISDN	4–6
4.1.4.3	Report parameters for CCS#7	4–7
4.1.5	Test suite execution control parameters	4–9
4.1.6	Log output parameters	4–10
4.1.7	Verdict log parameters	4–11
4.1.8	Prelliminary verdict parameters	4–12
5	**SOFTWARE INTERFACES TO MENU SYSTEM**	**5–1**
5.1	Interface to basic–software	5–1
5.2	Interface to TEST CASES.	5–2
6	**INDENTATION COMPILER**	**6–1**
6.1	Features of compiler	6–3
6.1.1	TEST STEP attachment	6–3
6.1.2	DEFAULT behaviour	6–4
6.1.3	EXIT TESTSTEP / CASE	6–6
6.1.4	RESCAN	6–6
6.1.5	REMAIN	6–7
7	**TEST STEP, CASE, GROUP, SUITE**	**7–1**
7.1	Define TEST CASES / STEPS	7–2
7.2	Create SIETEM tree structure	7–3
7.3	Attach the TEST SUITE to the MENU SYSTEM	7–4
8	**CASE SELECTION TOOL**	**8–1**

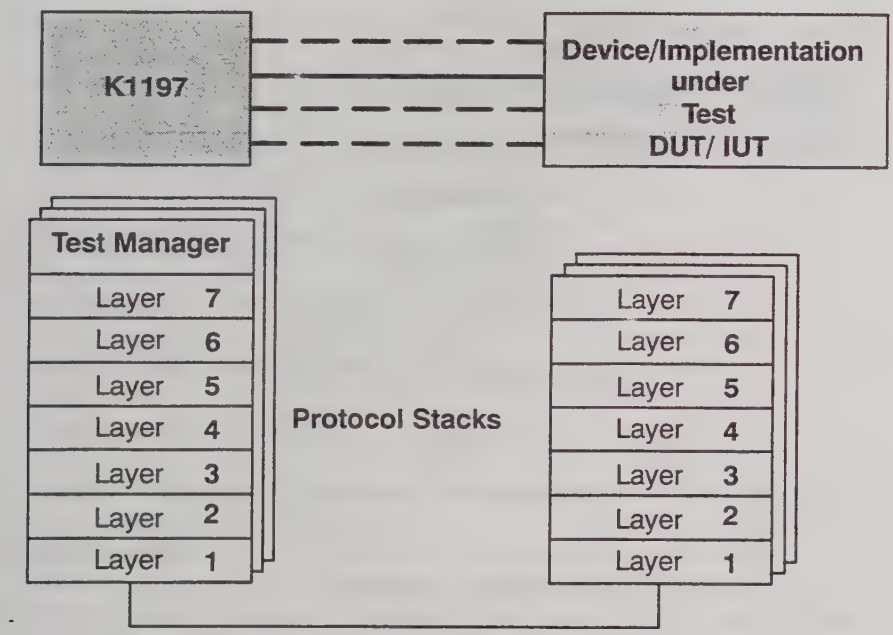
9	**VERDICTS**	**9–1**
9.1	Verdict words	9–1
9.2	TEST RUN	9–2
10	**TEST SUITE EXAMPLE**	**10–1**
11	**PICS/PIXIT PARAMETERS**	**11–1**
11.1	Define PICS/PIXIT–objects	11–2
11.2	Create a new MENU SYSTEM entry	11–4
11.3	Example	11–5
11.4	Words used by function: SAVEPARAM	11–5
11.5	' METHOD '	11–6
12	**OTHER WORDS**	**12–1**
12.1	Debugging words	12–1
12.1.1	' SPY '	12–2
12.2	Vectored output words	12–3
12.2.1	Select the output channel	12–4
12.3	TREE	12–5
12.4	'DOER' words	12–6
13	**SDL testmanager and SIETEM**	**13–1**
13.1	Overview	13–2
13.2	Enhancements for SIETEM	13–2
13.3	Source files on binary diskette	13–3
13.4	Example	13–6
INDEX		**I–1**

1 INTRODUCTION

SIETEM® means SIEmens TTCN TEst Manager

Is is designed for Acceptance / Conformance Tests with special interactive features.

Test Configuration:

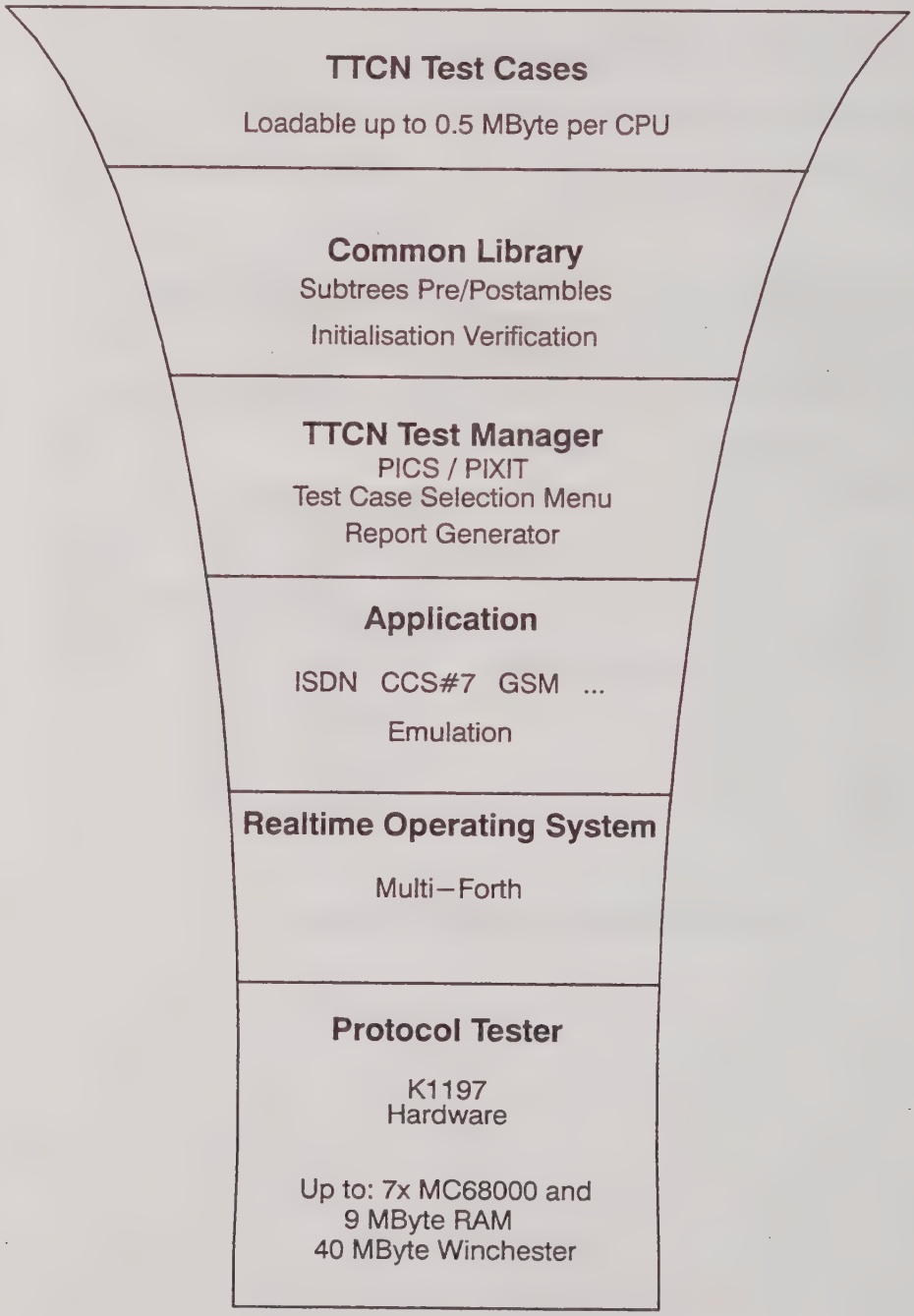


For further details see special manuals

SIETEM is a registered mark for Siemens TTCN Test Manager

TTCN is the **T**ree and **T**abular **C**ombined **N**otation, see *ISO/IEC DIS 9646–3*
SIETEM combines the advantages of the proven K1197 SDL Test manager with a TTCN subset.

1.1 SIETEM integration in K1197



1.2 Advantages of the 'SIETEM' implementation

Advantages of the SIETEM:

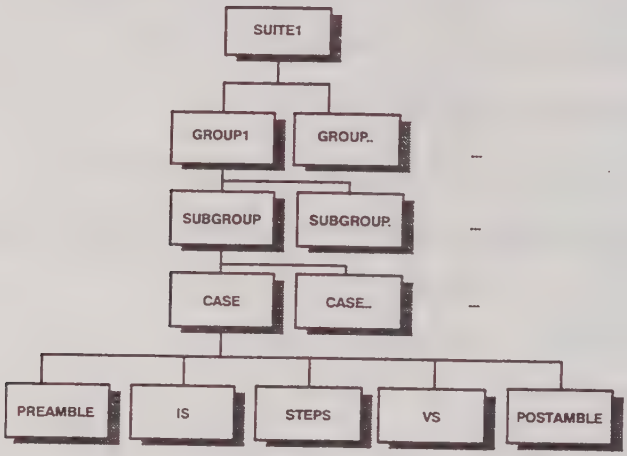
- The constructs may be mixed with the ones already available on K1197.
  (tested in numerous applications for DBP FTZ,CNET,CSELT,Telecom Australia etc.)
- All the available application programmes with their services may be used.
- Menu driven parameter selection and test case execution
- Hierarchical structure of test suites and unlimited use of subtrees and nesting.
- Verdicts, logs according to ISO 9646 Hierarchical structure of test case and display/ printout of the verdict data.
- Full verdict support (ISO 9646).
- Full test report support (ISO 9646)
- Full PICS, PIXIT support (ISO 9646)
- Default Behaviour (ISO 9646)
- The output functions may be directed to screen, trace, printer .. and combinations.
- Test case statistics
- The trace function reports/logs to capture buffer , screen , disk. The logs may be tra nsferred to disk, screen, printer.
- Provides a soft migration path towards TTCN.

Special supports for:

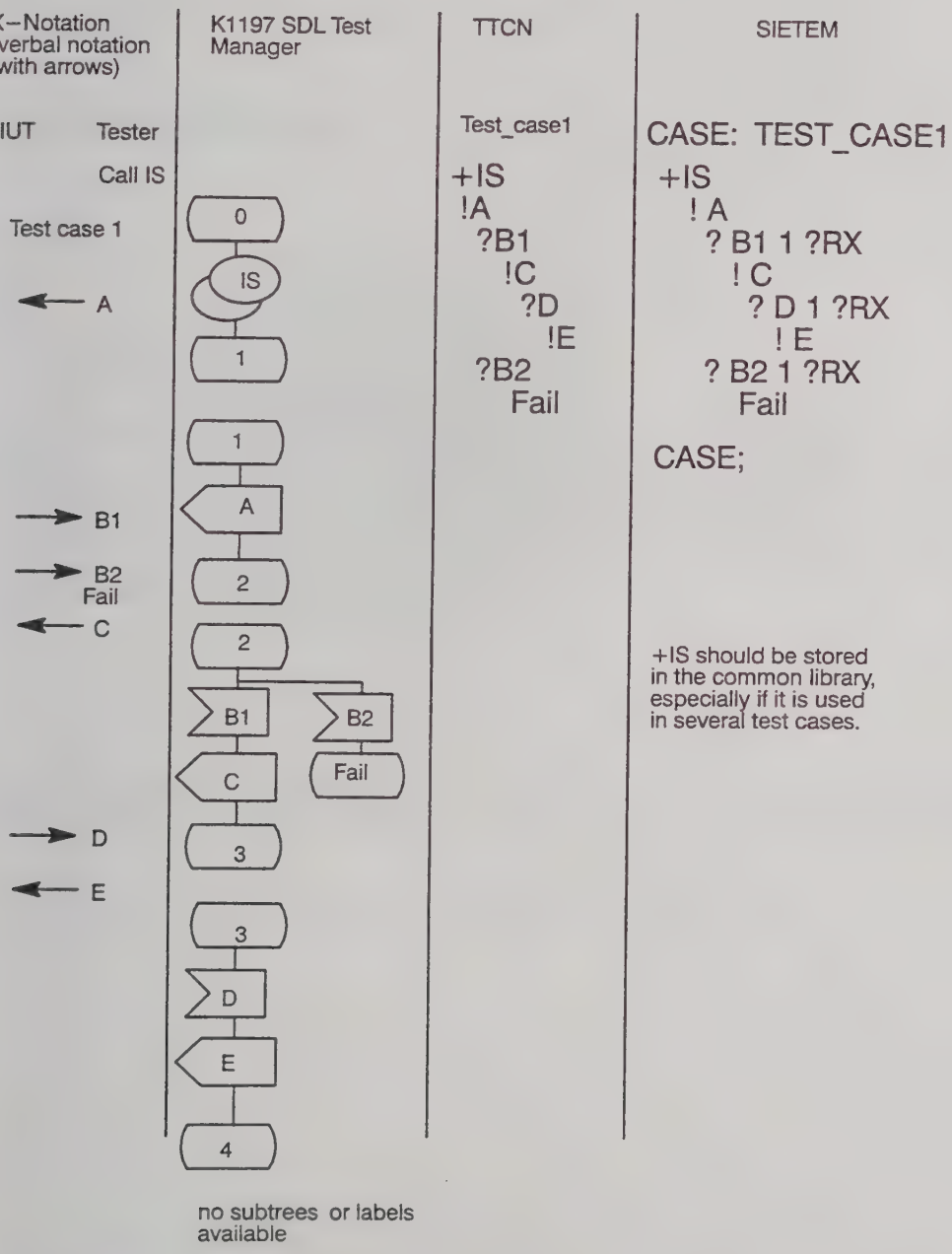
- Display of received event. (.RX)
- Calling of a subtree may be reported with full short and long name.
- All compiled trees and subtrees are kept in the dictionary and may be called in a tree by name.
  This starts the test manager and runs the tree or tree system.

1.3 TEST SUITE structure

The following picture shows the hierarchical structure of a test suite. A single test case may consist of an Initialization sequence (or Preamble), one or more test steps and a Verification sequence (or Postamble):



1.4 SIETEM in comparison with standard notation

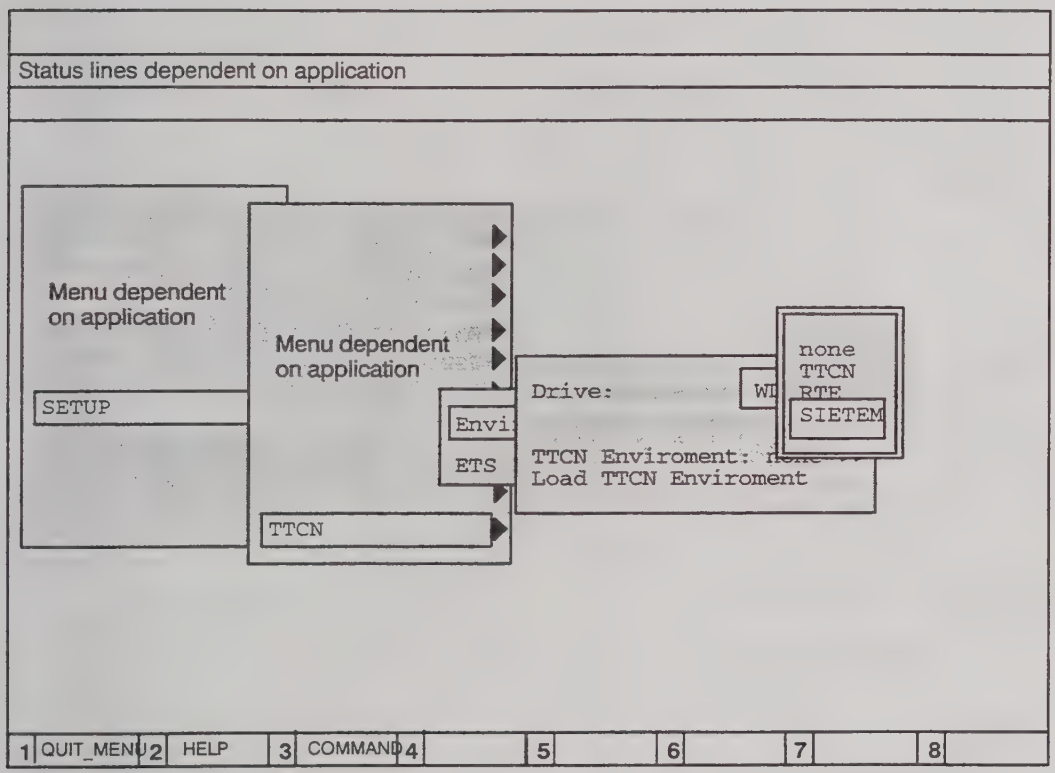


IS = Initialisation sequence

A growing number of specifications of acceptance– and conformance tests is written in TTCN

2 INSTALLATION

2.1 LOADING OF SIETEM SIETEM BINARY



In the menu system shown above (or other) you can load the SIETEM Binary.

2.1.1 ADAPTION OF THE STORAGE FOR CONFORMANCE TESTS UNDER SS NO.7

Following file, available on the binary disk, is an example, how to change the buffers of the K1197 for using large Conformance Tests:

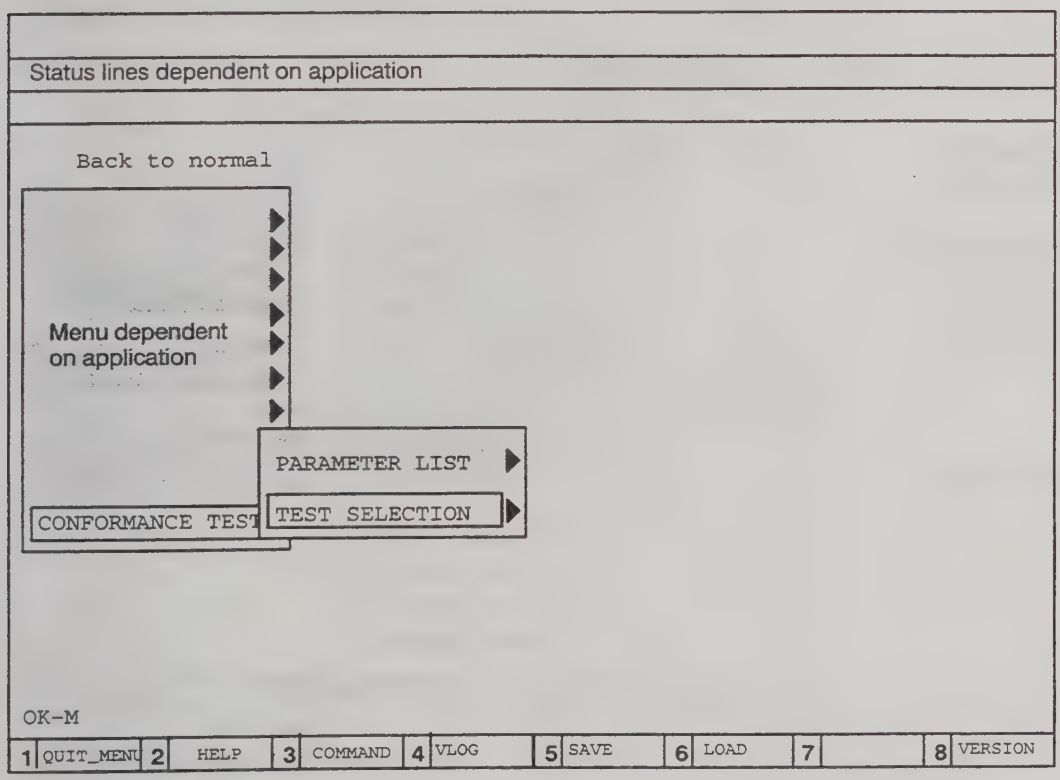
```
( ------------------------------------------------------ )
( FILE:      CONF_BUFFER.F                               )
( PURPOSE:   INCREASES BUFFERS FOR CONF TESTS            )
( DATE:      06.12.94                                    )
( ------------------------------------------------------ )
( ENDRAM = ENDRAM_M = 0X00100000 )
( ENDRAM_1 =          0X00E00000 from 0X00D00000 )
( ENDRAM_2 =          0X00D00000 from 0X00C00000 )
( ENDRAM_3 =          0X00C00000 from 0X00B00000 )
( ENDRAM_4 =          0X00B00000 from 0X00A00000 )
( ENDRAM_5 6 7 =      Nonexistent Processor 0X0010000 )
( RTEEND RTEEND_1 ... COLD HERE without MENU )
( string table in CPU1 from $MStrStart @ )
( MALLOC in CPU1 with pointer MEXT-END @ )
( EXT.RAM for CPU1 )
( EDITBUF in main )
( DICTLIMIT @ , HERE .RAM      )
( ------------------------------------------------------ )
( ------------------------------------------------------ )
( ================================= )
( +------------+                     )
( |            | 0XB00000 ENDRAM_4   )
( |            |                     )
( | Verd. Buff |                     )
( |            |                     )
( |    .       | 0XABEC00 S-B @ .H   )
( +------------+                     )
( |            | 0XABEC00 h-end .H   )
( |            |                     )
( | hash-table |                     )
( |            | 0XAA0000            )
( |            |          used!      )
( |            | 0XA7EC00 h_tab .H   )
( +------------+                     )
( |            |                     )
( .            .                     )
( .            .                     )
( |            | 0XA00000            )
( +------------+                     )
( ------------------------------------------------------ )
( ------------------------------------------------------ )
```

```
( ------------------------------------------------ )
( Move the Verdict-Buffer to CPU4                  )
( Buffer-Size: VERD_SIZE kByte                     )
( ------------------------------------------------ )
261 KBYTE CONSTANT VERD_SIZE
: VERDICT_BUF
    ENDRAM_4 VERD_SIZE - S-B !
    VERD_SIZE S-B-SIZE !
    INIT_CAPT
    CR
    ." VERDICTLOG with " VERD_SIZE 10 >># . ."  KB beginning at: " S-B @ .H CR
;
( ------------------------------------------------ )
( Move the hadh-table to CPU4 - VERD_SIZE          )
( Buffer-Size: h-max     kByte                     )
( ------------------------------------------------ )
( Can be set to 192, maybe ?! )
256 KBYTE              CONSTANT h-max
ENDRAM_4 VERD_SIZE -   CONSTANT h-end  ( Dependent on Verdict Buffer! )
: .HASH
    ." HASH   Start=" h_tab .H ."  End=" h-end .H
    ."   Size=" h-max 10 >># . ." KB  "    CR
;
OFF QUICKY
: HASH-PATCH
    ." Setup HASH-table to CPU4" CR
    h-end h-max  -    ' h_tab !
    h_tab h_size +    ' he_top !
    he_top 4+         ' he_stack !
    .HASH
;
MAKE <INI-PATCH>
   VERDICT_BUF
   HASH-PATCH
;
<INI-PATCH>
ON QUICKY
( ------------------------------------------------------ )
( ------------------------------------------------------ )
```

3 MENU SYSTEM

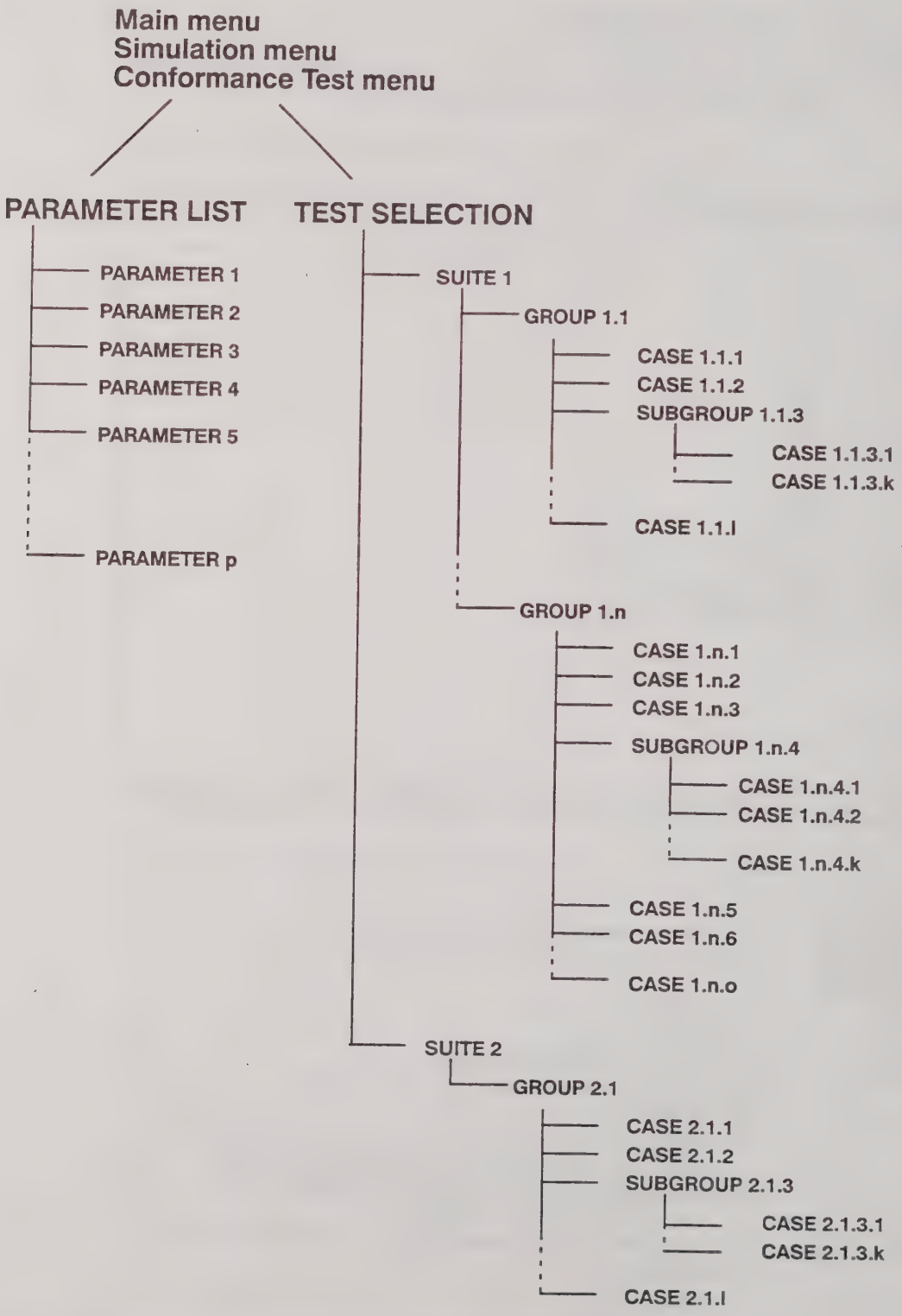
3.1 Parameter and test selection menu

After loading an application this menu is available.
Parameter setting as well as test case selection and execution is menu driven. The Main Menu item 'CONFORMANCE TEST' leads (press cursor right) to the following submenu:



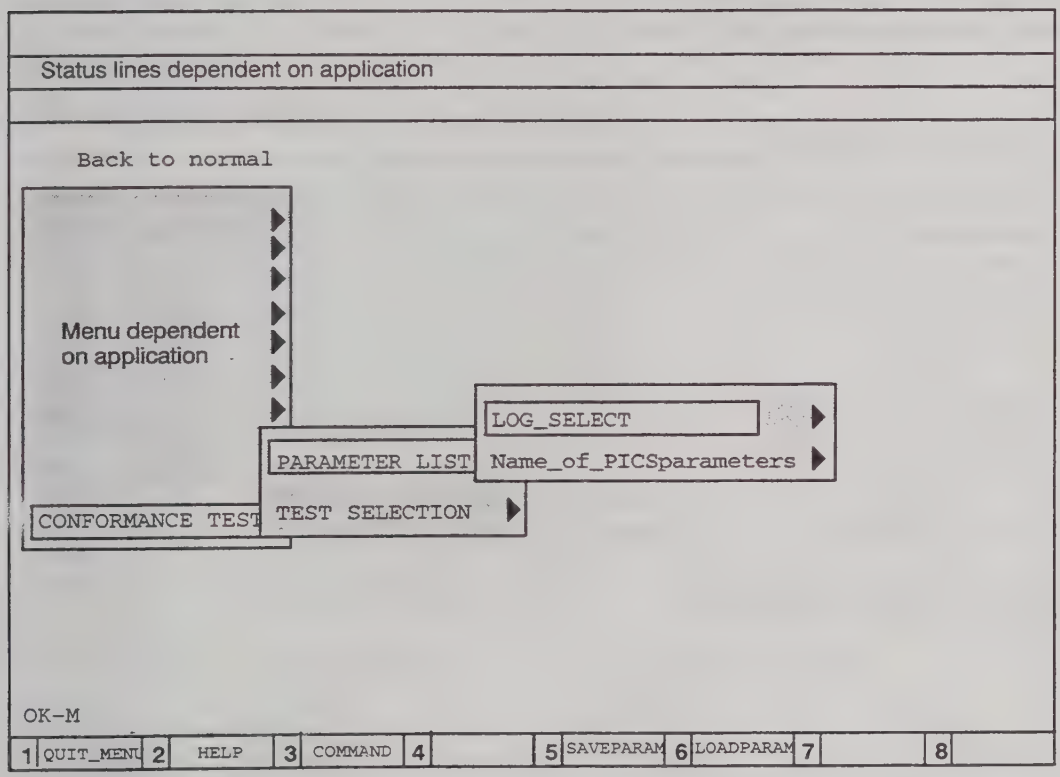
MENU SELECTION:	
PARAMETER LIST	: List of the PICS and PIXIT variables
TEST SELECTION	: Test Cases tree to be selected and executed
FUNCTION KEYS:	
QUIT_MENU	: Close the menu window
HELP	: Open a help window
COMMAND	: Open FORTH–Interpreter window
VLOG	: Show a list of Verdict of Test Cases wich are finished (See Chapter 3.2.1)
SAVE	: Save the actual values of all PICS/PIXIT parameters or test cases.
LOAD	: Load previously saved PICS/PIXIT parameters or test cases. **Note**: Do not use key *DEL* to delete characters of the name of the PICS/PIXIT list! Use key *BS* intead.
VERSION	: Shows the version of the loaded software in the command window (See Chapter 12.4)

EXAMPLE OF A CONFORMANCE TEST MENU:



3.1.1 Parameter menu

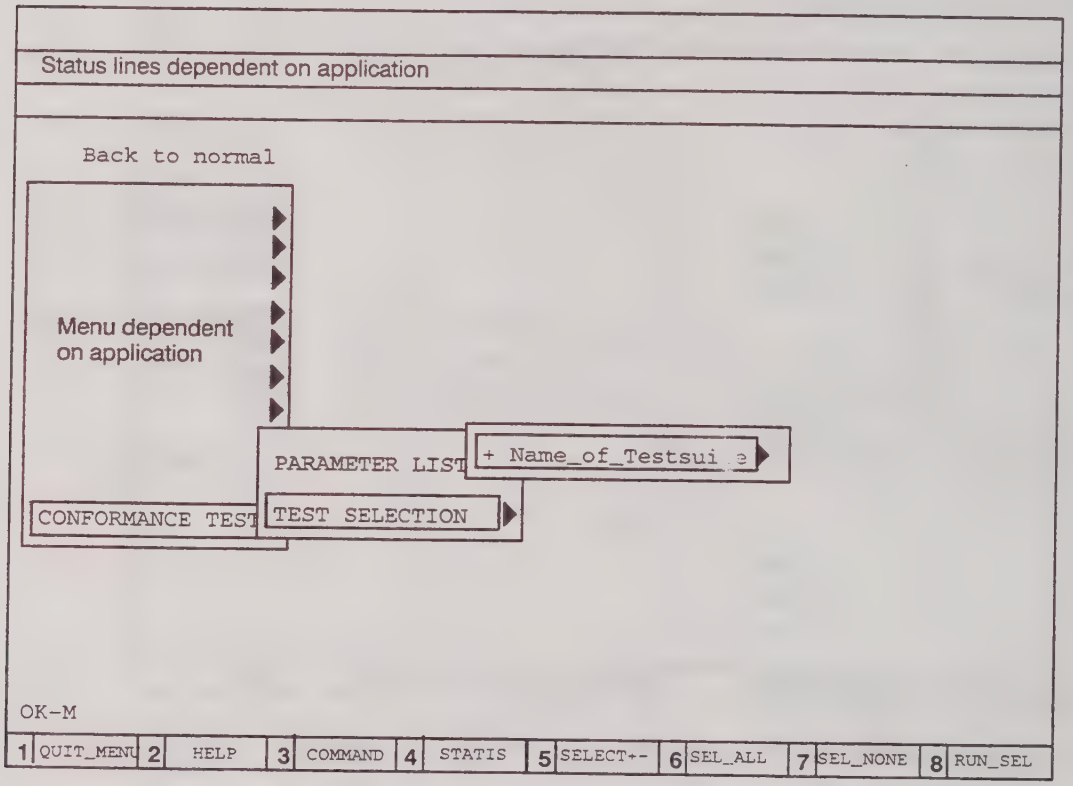
This menu leads into the submenus of several parameter tables. The LOG SELECT table contains SIETEM parameters, which will be described below. The PICS and PIXIT parameter tables are dependent of the actual Conformance Test Suite.



MENU SELECTION:	
LOG_SELECT	: SIETEM Parameters (see chapter 4)
FUNCTION KEYS:	
QUIT_MENU	: Close the menu window
HELP	: Open a help window
COMMAND	: Open FORTH–Interpreter window
SAVE_PARAM	: Save the actual values of the selected PICS/PIXIT parameter
LOAD_PARAM	: Load previously saved PICS/PIXIT parameters

3.1.2 Test case selection menu

The test selection menu contains submenus of one or more test suites. Each test suite menu consists of several submenus containing the groups, subgroups and cases. Each entry consists of a text concerning the test purpose and the name of the suite/group/subgroup/case. A leading plus means this suite/group/subgroup/case is selected for running, a minus means not. Each test case entry ends with the latest verdict value of this test case. Capital letters mark verdicts, which are the result of the test sequence, small letters mark verdicts, which are assigned by the operator.



FUNCTION KEYS:

QUIT_MENU	: Close the menu window
HELP	: Open a help window
COMMAND	: Open ORTH-Interpreter window
STATIS	: Shows a statistic about the Test Cases wich are finished (See Chapter 3.2.2)
SELECT+-	: Selecting or deselecting of items to be run or not when invoked
SEL_ALL	: Selecting all items of the current suite or group.
SEL_NONE	: Deselecting all items of the current suite or group.
RUN_SEL	: Running the current selected suite, group, subgroup or case, where the green cursor points to.

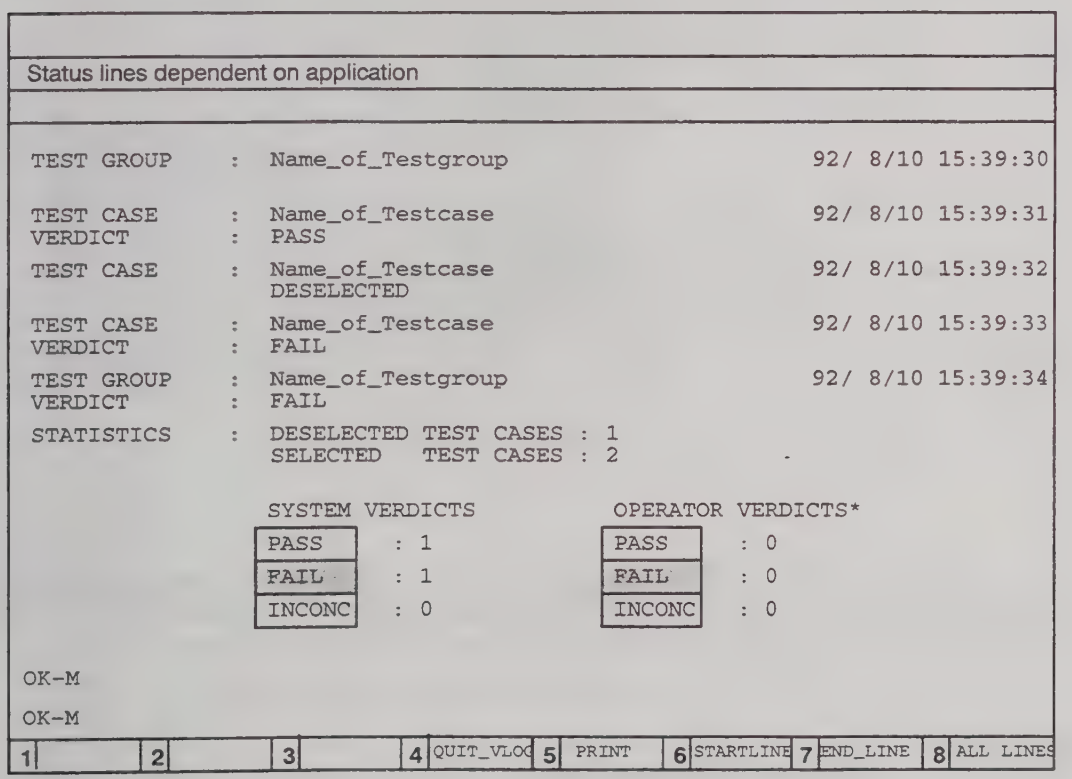
When a test case / group / suite is deselected with **f5**, the name will be shown in white letters (instead of light blue letters when it is selected) and there is a – instead of a + in front of the name.

3.2 Special functions

3.2.1 VERDICT LOG (VLOG)

For each test case/subgroup/group/suite an entry will be made in the Verdict Log, if enabled in the LOG SELECT menu. This entry consists of the name and purpose text and the verdict, if the test was executed. Deselected tests are marked as such. Furthermore each operator verdict and operator comment is inserted in the Verdict Log buffer. Also it is possible to get a statistic entry per subgroup, group or suite, if enabled via LOG SELECT parameter.

To get this function see Chapter 3.1.



FUNCTION KEYS:

QUIT_VLOG	: Close the VLOG window
PRINT	: Print the contents of the Verdict Log.
STARTLINE	: Mark the line at the current cursor position as first line to be printed.
END_LINE	: Mark the line at the current cursor position to be printed as last line.
ALL_LINES	: Mark the whole Verdict Log buffer to be printed.

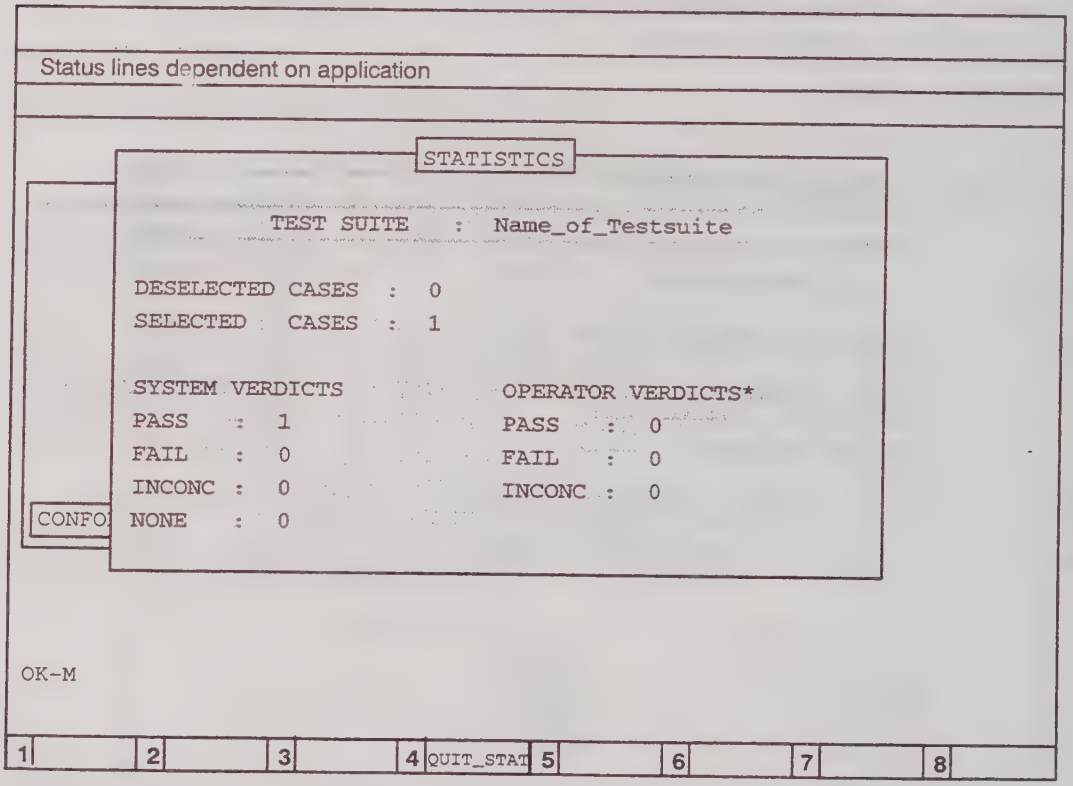
Inside this menu the cursor keys can be used to display and scroll the Verdict Log buffer contents.

* This function is not available on CCS#7

3.2.2 Statistics

The statistic function of a suite/group/subgroup gives an overview of the number of selected and deselected test cases. For the selected test cases also an overview of verdicts and operator verdicts is shown.

To get this function see Chapter 3.1.2.

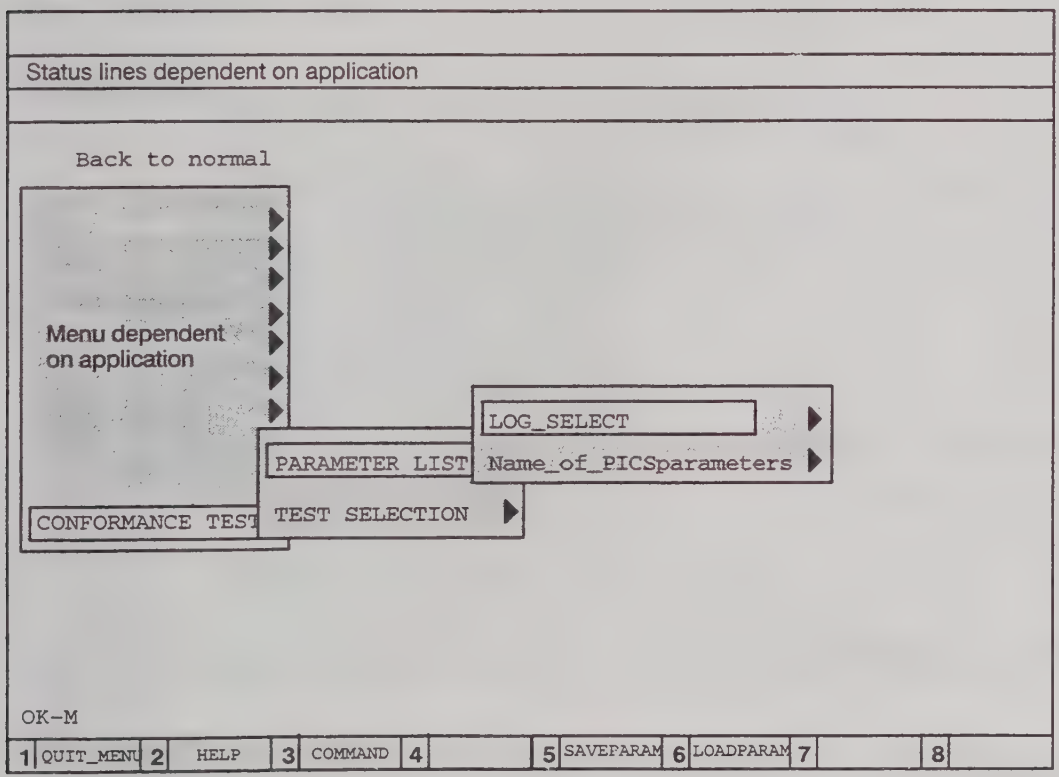


FUNCTION KEYS:
QUIT_STAT : Close the STATISTICS window

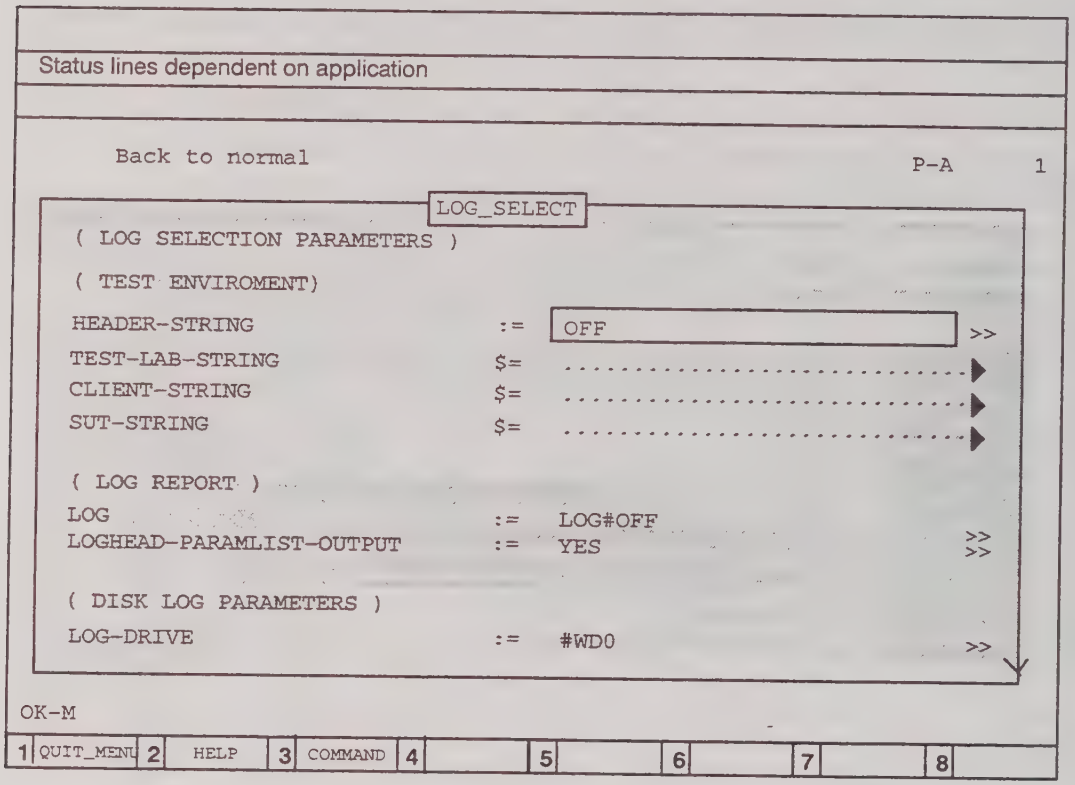
* This function is not available on CCS#7

4 SIETEM parameters

To change the values of the SIETEM parameters, please move with the cursor keys to the following menu:



Then move right with the cursor to following menu:



The standard access to PICS and PIXIT tables is embedded in the menu system. Selecting the Parameter (PICS/PIXIT) menu pages displays the parameters with its actual values as contained in the variables. That is the default setup. If a parameter has to be changed, it has to be selected on the menu page via cursor keys.

4.1 LOG_SELECT

LOG SELECTION PARAMETERS

4.1.1 TEST ENVIROMENT

HEADER–STRINGS*

Possible Values: ON **OFF**

Meaning:

Prints a LOG–Header.

ON: Following strings will be print as header of the LOG–file.

OFF: Following strings will not be print as header of the LOG–file.

TEST–LAB–STRING*

Meaning:

Prints information about the Test laboratory in the LOG–file.

CLIENT–STRING*

Meaning:

Prints information about the client (tester) in the LOG–file.

SUT–STRING*

Meaning:

Prints information about the 'System Under Test' in the LOG–file.

Note: The strings shall not contain spaces (0x20)!

4.1.2 LOG REPORT PARAMETERS

LOG

Possible values: **LOG#OFF**

LOG#CASE

Meaning:

No LOG–REPORT is written, or after each test case the LOG–REPORT is written.

LOGHEAD–PARAMLIST–OUTPUT

Possible values: **YES** NO

Meaning:

If YES is used, the complete actual status of all parameter/PICS/PIXIT lists is output at the start of a test log.

* Only available for CCS#7–Application

4.1.3 DISK LOG parameters

LOG–DRIVE

Possible values: #DR0
#DR1
#WD0
#WD1
#WD2
#WD3
#WD4
#WD5
#WD6
#WD7

Meaning:
Drive the DISK–LOG files should be written.
Note:
Use no **DD**–(Double Density)–Disks!

DISKLOG_PASS

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are stored to disk, where test cases name is file name in LOG–REPORT.

DISKLOG_FAIL

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are stored to disk, where test cases name is file name in LOG–REPORT.

DISKLOG_INCONC

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are stored to disk, where test cases name is file name in LOG–REPORT.

ONE_FILE_PER_CASE/GROUP

Possible Values: YES **NO**
Meaning:
Applies only for disk log.
NO: The complete suite or part of it is logged into one file.
YES: Each case or group log is put into a separate file. Only the long format creates a file.

4.1.4 PRINT LOG parameters

PRINTLOG_PASS

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are written in LOG–REPORT, in case the test ends with verdict "PASS".

PRINTLOG_FAIL

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are written in LOG–REPORT, in case the test ends with verdict "FAIL".

PRINTLOG_INCONC

Possible values: **LOG#OFF**
LOG#SHORT
LOG#LONG

Meaning:
Nothing / name and verdict / name, trace and verdict are written in LOG–REPORT, in case the test ends with verdict "INCONC".

4.1.4.1 REPORT PARAMETER FOR X.25

PRINT–REPORT–TYPE

Possible values: **SHORT#REPORT**
COMPLETE#REPORT
HEX#REPORT
CHAR#REPORT

Meaning:
With this parameter, different report types may be selected for screen display of layer 4 events and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

4.1.4.2 Report parameters for ISDN

PRINT-L3-REPORT-MODE

Possible values: **L3SHORT#MODE**
L3MEDIUM1#MODE
L3MEDIUM2#MODE
L3COMPLETE#MODE

Meaning:
With this parameter, different report types may be selected for screen display of layer 3 events and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT-L3-REPORT-TYPE

Possible values: L3_NONE#REPORT
L3_MNEMONIC#REPORT
L3_HEX#REPORT
L3_CHAR#REPORT

Meaning:
With this parameter, different report types may be selected for screen display of layer 3 events and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT-L2-REPORT-TYPE

Possible values: L2_NONE#REPORT
L2_COMPLETE#REPORT
L2_HEX#REPORT
L2_CHAR#REPORT
L2_SHORT#REPORT

Meaning:
With this parameter, different report types may be selected for screen display of layer 2 events and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT-L1-REPORT-TYPE

Possible values: L1_NONE#REPORT
L1_COMPLETE#REPORT
L1_SHORT#REPORT

Meaning:
With this parameter, different report types may be selected for screen display of layer 1 events and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

4.1.4.3 Report parameters for CCS#7

PRINT–L4REPORT–TYPE

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#HEX
PRINT#CHAR
PRINT#COMP
PRINT#MED
PRINT#MED2
PRINT#MED3
PRINT#MED4

Meaning:
With this parameter, different report types may be selected for screen display and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT–TCAPREP–TYPE*

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#HEX
PRINT#CHAR
PRINT#COMP
PRINT#MED
PRINT#MED2
PRINT#MED3
PRINT#MED4
PRINT#MED5

Meaning:
With this parameter, different report types may be selected for screen display and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT–SCCPREP–TYPE

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#HEX
PRINT#CHAR
PRINT#COMP
PRINT#MED
PRINT#MED2
PRINT#MED3
PRINT#MED4

Meaning:
With this parameter, different report types may be selected for screen display of SCCP message and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT–L3REPORT–TYPE

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#HEX

* Only available if TCAP–User Part is used

PRINT#CHAR
PRINT#COMP
PRINT#MED
PRINT#MED2
PRINT#MED3
PRINT#MED4

Meaning:
With this parameter, different report types may be selected for screen display and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT-L2REPORT-TYPE

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#HEX
PRINT#CHAR
PRINT#COMP
PRINT#MED
PRINT#MED2
PRINT#MED3
PRINT#MED4

Meaning:
With this parameter, different report types may be selected for screen display and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

PRINT-L1REPORT-TYPE

Possible values: **PRINT#OFF**

PRINT#SHRT
PRINT#COMP

Meaning:
With this parameter, different report types may be selected for screen display and for printout. The screen report type is selected via monitor/ report menu. The print log report type is taken from this parameter.

4.1.5 Test suite execution control parameters

RUN–SELECTION–MODE

Possible values: **ALL** ONLY–CASES–WITH–VERDICT<>PASS

Meaning:

IF ONLY–CASES–WITH–VERDICT<>PASS is used, only those test cases will be executed, which have no PASS Verdict got before.

RESTRICTED_CASES/GROUPS/SUITES

Possible values: **ON** OFF

Meaning:

If OFF is used, all the invoked cases are run, no matter if they are selected (appear green in the case selection menu) or not. If ON is used, only the selected (green) test cases/groups/ subgroups/suites are run.

REGULATORY_CAMPAIGN

Possible values: **ON** OFF

Meaning:

If ON is used, only the PICS dependent selected test cases/groups/ subgroups/suites are or can be enabled.So it is assured, that only those test cases, that make sense in the light of the PICS, are run.

WAIT_TIME_AFTER_CASE

Possible values: [0 .. 10000]

Meaning:

Time between two running test cases in a group, subgroup or suite, where you can check the last test, or where you can type in FORTH commands.

4.1.6 Log output parameters

CASE_SEPARATOR

Possible values: **HORIZONTAL-BAR**
HORIZONTAL=BAR
FORM#FEED
OFF

Meaning:
Line Separator (------ , ====== or FF) for printer in LOG-REPORT.

GROUP_SEPARATOR

Possible values: **HORIZONTAL=BAR**
FORM#FEED
OFF
HORIZONTAL-BAR

Meaning:
Line Separator (------ , ====== or FF) for printer in LOG-REPORT.

CGS_HEADER_DATE

Possible values: **ON** OFF

Meaning:
If ON is used, each test case/group/.. in the log will output the date. If there is sufficient space on the line, also the time of day is included.

LINE_AFTER_VERDICT

Possible values: **ON** OFF

Meaning:
If ON is used, the verdict capture entry will be followed by a new line.

4.1.7 Verdict log parameters

VERDICT-LOG

Possible values: **YES** NO
Meaning:
If YES is used, the Verdict Log will be allocated

VERDICT-LOG-PASS-IMMED

Possible values: YES **NO**
Meaning:
If YES is used, the respective verdicts are written immediately to the Verdict Log

VERDICT-LOG-FAIL-IMMED

Possible values: YES **NO**
Meaning:
If YES is used, the respective verdicts are written immediately to the Verdict Log

VERDICT-LOG-INCONC-IMMED

Possible values: YES **NO**
Meaning:
If YES is used, the respective verdicts are written immediately to the Verdict Log

VERDICT-STATISTICS-SUBGROUP

Possible values: **YES** NO
Meaning:
If YES is used, a statistic of the respective suite/group/subgroup is inserted in the Verdict Log.

VERDICT-STATISTICS-GROUP

Possible values: **YES** NO
Meaning:
If YES is used, a statistic of the respective suite/group/subgroup is inserted in the Verdict Log.

VERDICT-STATISTICS-SUITE

Possible values: **YES** NO
Meaning:
If YES is used, a statistic of the respective suite/group/subgroup is inserted in the Verdict Log.

4.1.8 Prelliminary verdict parameters

PASS–IMMED–DISPLAY

Possible values: **NO** YES

Meaning:

If YES is used, the respective verdicts are output immediately. If NO is selected, the verdicts appear only as final verdicts at the end of the test case/subgroup/group/suite.

FAIL–IMMED–DISPLAY

Possible values: NO **YES**

Meaning:

If YES is used, the respective verdicts are output immediately. If NO is selected, the verdicts appear only as final verdicts at the end of the test case/subgroup/group/suite.

INCONC–IMMED–DISPLAY

Possible values: NO **YES**

Meaning:

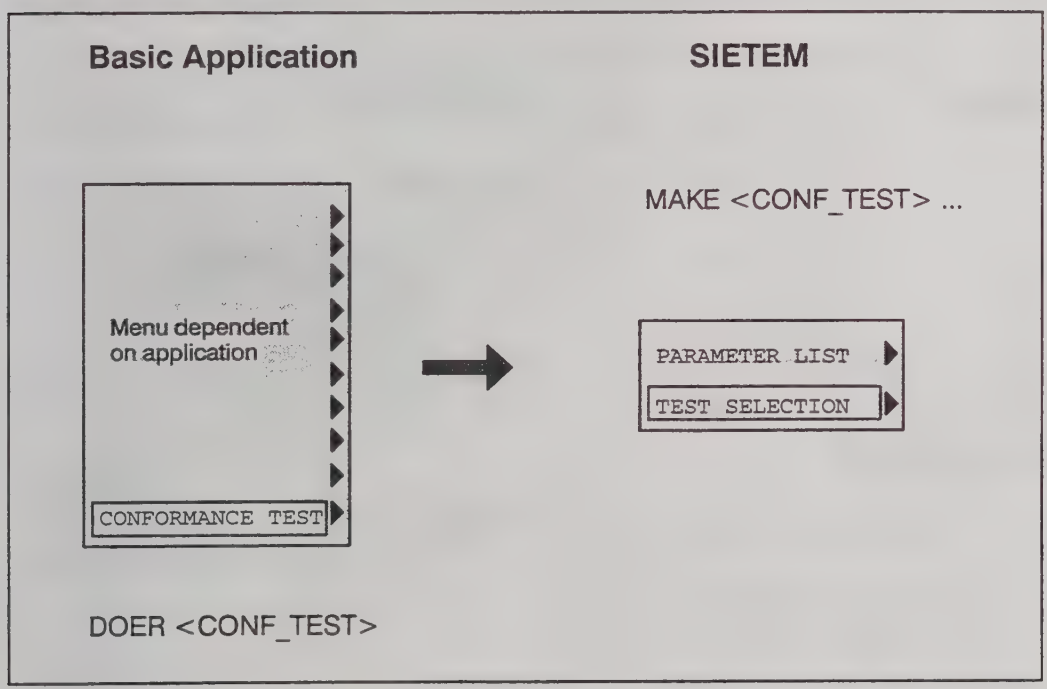
If YES is used, the respective verdicts are output immediately. If NO is selected, the verdicts appear only as final verdicts at the end of the test case/subgroup/group/suite.

5 SOFTWARE INTERFACES TO MENU SYSTEM

5.1 Interface to basic–software

The interface between SIETEM and the Basic–Application consists of the DOER word **<CONF_TEST>** which will be executed when the menu item CONFORMANCE TEST will be called.

In SIETEM a 'MAKE' to <CONF_TEST> will be done, so the first SIETEM menu appears by calling the menu item CONFORMANCE TEST.

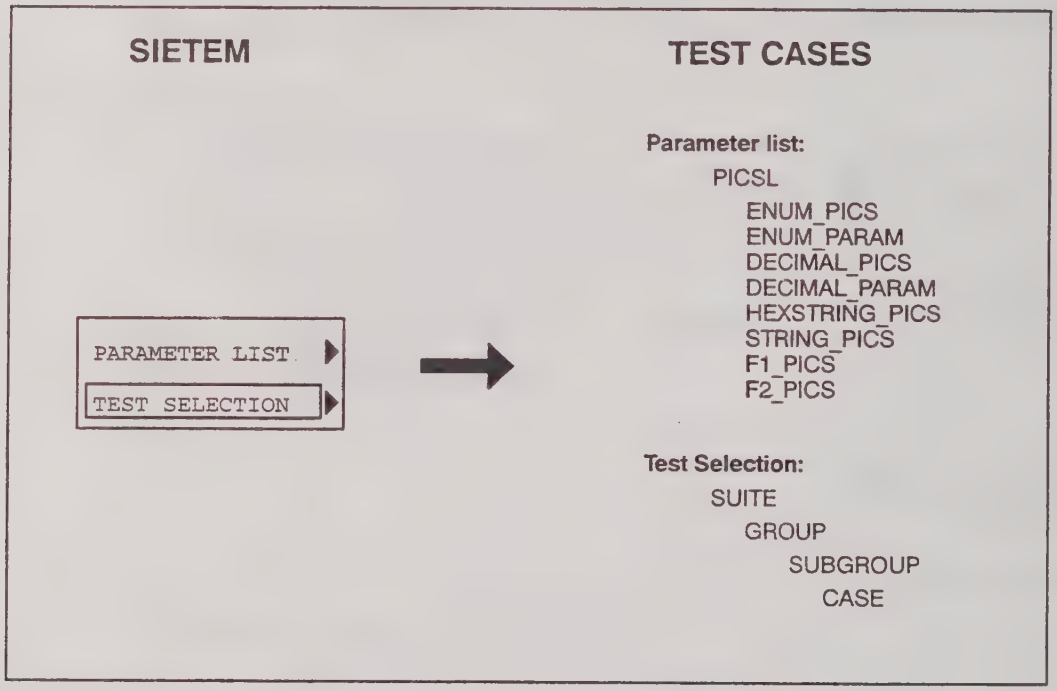


5.2 Interface to TEST CASES.

The interface between SIETEM and the test cases consist of the definition of:

STEP and **PICSL**
CASE
SUBGROUP
GROUP
SUITE

The output of the names of the testcases and other texts will be put into menus. The layout of the menus is determined, but the size of it will be set flexible during compiling time of the test cases. It depends on the number of SUITE, GROUP, .. , PICSL definitions.



For more information see chapters 7 and 11 .

To allow the ETS binary to run well a so called "INIT" file is needed. The file contains the definition of the word **TTCN_SUITE_INIT** that is called every time the ETS binary is loaded. The word `TTCN_SUITE_INIT` initializes the system to run the ETS (Connects test cases and parameters to menu system) .

Example:

```
: TTCN_SUITE_INIT  ( -- )
    AUTO_SET_PARAM SUITE-PICSL-1
    AUTO_SET_PARAM SUITE-PICSL-2
   AUTO_SET_PARAM SUITE-PICSL-3        ( SET THE TEST SUITE PARAMETER MENUS )
    AUTO_SET_SUITE ISDN3                 ( SET THE TEST CASE MENU )
;
```

6 INDENTATION COMPILER

ON / OFF ATCM switches the compiler on / off

This compiler changes, when switched on, the behaviour of the normal FORTH compiler. The FORTH compiler becomes indentation–sensitive, as necessary for TTCN.
It reacts on (column indented) **?** symbols, followed by a space, at the beginning of a line.

With this possibility of identation you can create sequential parts and alternatives of receiving messages in a short and well structured way:

Sequential usage:

```
! Send_msg1
   ? Receive_msg2
      ! Send_msg3
         ? Receive_msg4
              .
              .
```

Meaning:

First send message1, then wait for receiving message2, after that send message3 and wait for receiving message4, and so on.

Usage of alternatives:

```
! Send_msg1
   ? Receive_msg2
        ! Action2_1
   ? Receive_msg3
        ! Action3_1
   ? Receive_msg4
        ! Action4_1
              .
              .
              .
```

Meaning:

First send message1, then wait for receiving message 2 or message 3 or message 4.
When, for example, message3 is received, the Action3_1 will be done. In this case Action2_1 and Action4_1 **never** will be executed.

Note: While 'ON ATCM' do **not** use any kind of colours or tabs in the editor of the K1197 !

? **(--)**
TTCN Receive Operator

! **(--)**
TTCN Send Operator

Note: The behaviour of the word '!' depends on the ATCM mode:
OFF ATCM:
! : FORTH 'store' Operator
ON ATCM:
! : TTCN 'Send' Operator
L!: FORTH 'store' Operator

IF? **(flag --)**

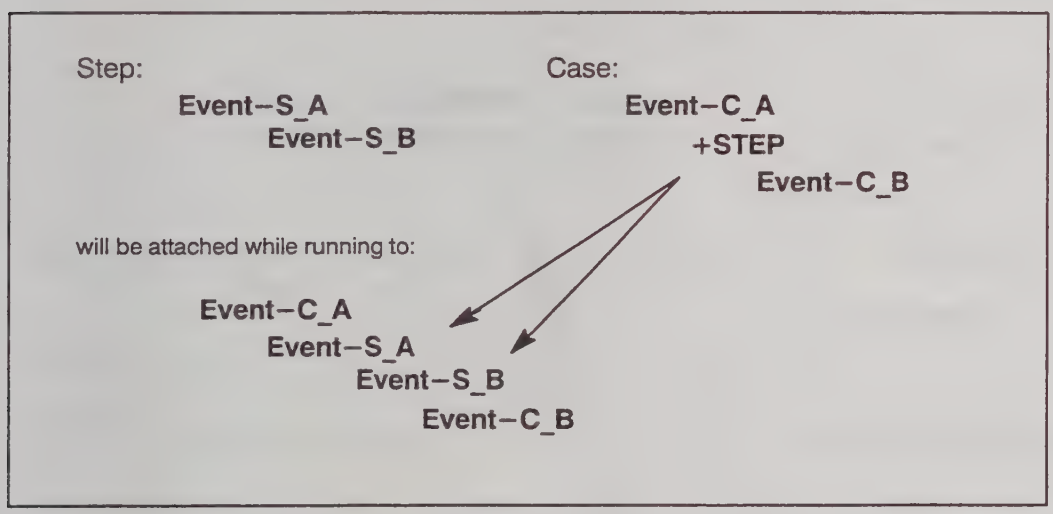
'IF' construct, depending on the occurrence of '!' or '?'.
When after '?', execute **IF part**
When after '!', execute **ELSE part.**

6.1 Features of compiler

There are some special features according to ISO/IEC–DIS 9464–3 used in SIETEM.

6.1.1 TEST STEP attachment

Some sequences of sending and receiving messages will be used in several test cases. For this usage, you can create a test step (see chapter 7.1). This test step will then be called by the test cases:



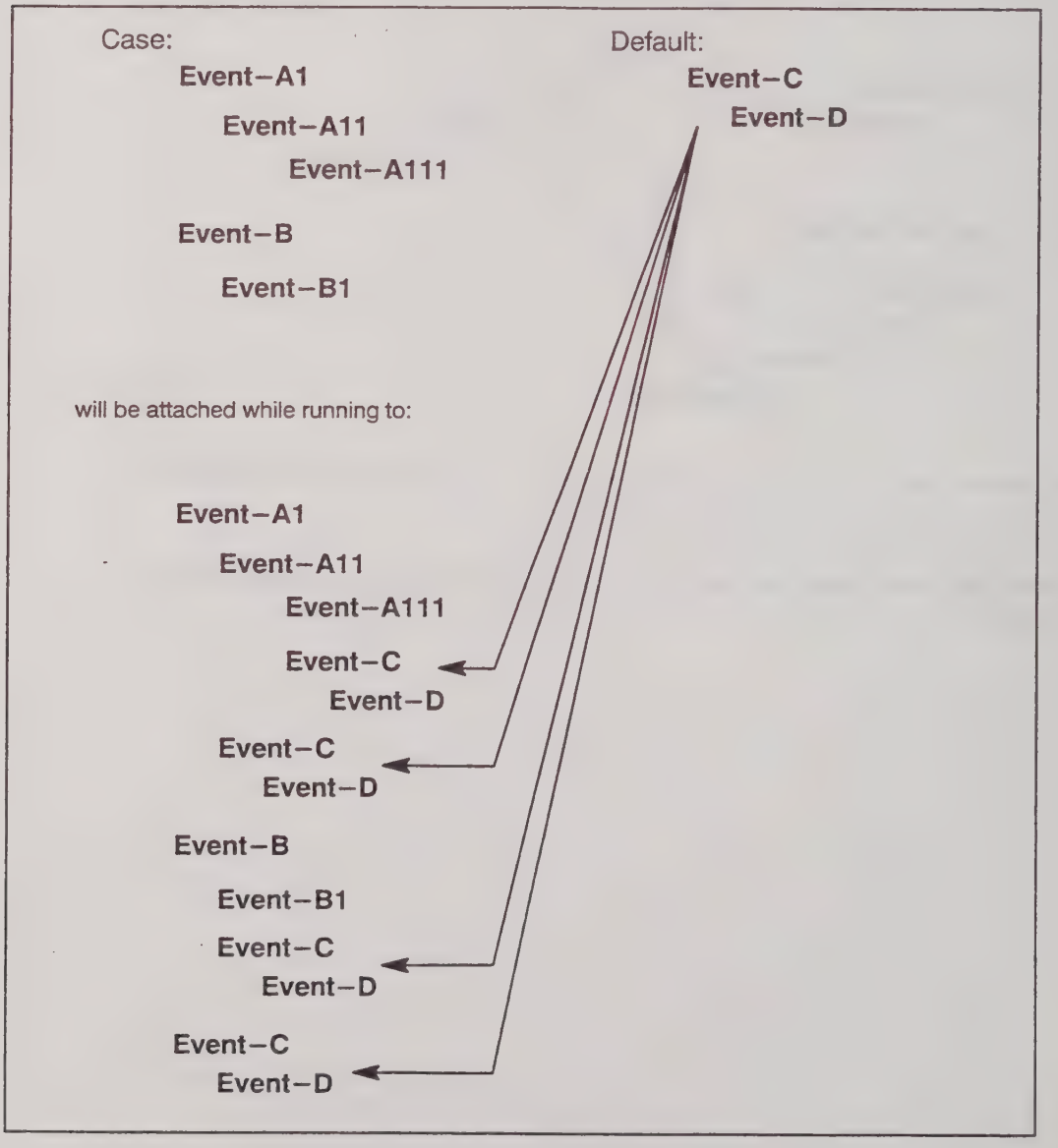
The name of a test step should always begin with a '+', then an attached test step looks like defined in ISO/IEC–DIS 9464–3.

6.1.2 DEFAULT behaviour

A TTCN test specification shall specify alternative behaviour for **every** possible event. It often happens that in a behaviour tree every sequence of alternatives ends in the same behaviour. This behaviour may be 'factored out' as default behaviour to this tree. Such default behaviour descriptions are located in the global defaults library.

In many cases the default construct will be used to emphasize a set of 'interesting' paths through a test by declaring the less interesting common alternatives as default behaviour.

DEFAU Γ ATTACHMENT:



SPECIAL WORDS:

DEFAULT: (--)
Start of a default definition
Usage:
```
DEFAULT: name_of_default
   .
   .
DEFAULT;
```

DEFAULT; (--)
End of a default definition

DEFAULT= (--)
Identifies Default structure for all following test cases
Usage: `DEFAULT= name_of_default`

Note: To switch off default behaviour type:
`DEFAULT= NOTHING`

Example:

```
ON ATCM
DEFAULT: Name_of_DEFAULT-part
         ? . . .
         .
         .
         .
DEFAULT;

DEFAULT= Name_of_DEFAULT-part

OFF ATCM
```

6.1.3 EXIT TESTSTEP / CASE

In case of a final FAIL or an unexpected event in a test step / case it makes no sense to go on. So you can break off the test step / case and leave it.

Note: Use these words only in a test case / step, usually after calling the postamble or in default behaviour!

ULTIMATE_FAIL **(--)**
Outputs the name of the test step or contraint where the test case is broken down.

!SHUTDOWN **(--)**
Initiates shutdown of a test case / step.

?SHUTDOWN **(-- flag)**
Shutdown the test case / step and ends all running emulations in a correct way. This word should always be called after a **?** in receiving line.

EXIT-> **(--)**
Leaves the test step / case in a correct manner.

Example:

```
ON ATCM
CASE: Case_name
          .
          .          .
       ? Timer_No ?TIMER
           ULTIMATE_FAIL
           (FAIL)
           !SHUTDOWN
             ? ?SHUTDOWN
               EXIT->
CASE;
OFF ATCM
```

6.1.4 RESCAN

RESCAN causes SIETEM on an event that did not match the event list nor the default behaviour's event list to drop the event, wait for a new one and start again the event matching process at the beginning (first '?' in column) of the current level. (See DIS ISO 9646)

YES_LOOP! **(--)**
Enables Rescan. (Default value)

NO_LOOP! **(--)**
Disables Rescan.

6.1.5 REMAIN

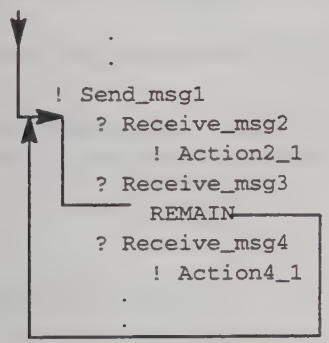
REMAIN enables to jump backwards in test steps / cases.
'REMAIN' is identical to the 'RETURN' statement of the further ISO/IEC–IS 9464–3.

Note: Use this feature only with caution!
It destroys the sequential structure of a test case.
Please avoid this feature, if possible!

REMAIN **(– –)**
Compiler jumps back to the first '?' in the column REMAIN was called.

REMAIN_DEFAULT **(– –)**
Like REMAIN, but used in ' DEFAULT'–Construct.

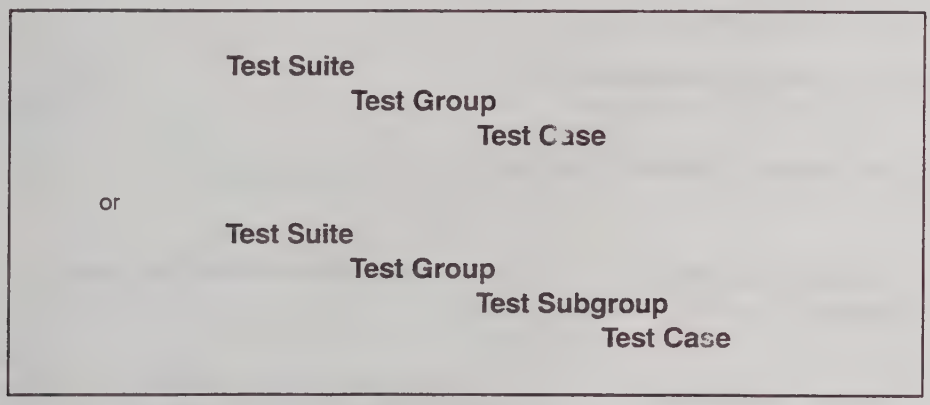
Example:



In this example the message3 was received, so the test case will remain in this column and start again with waiting for a message.

7 TEST STEP, CASE, GROUP, SUITE

The structure of a conformance test suite is mainly, according to ISO 9646:



But even totally flat structures with only cases in a suite are possible.

Note: Recursive calls between suite–group–case are **not** allowed!

The number of entries is unlimited for GROUP, SUBGROUP and CASE. The limit for SUITE is 10 entries.
STEP should not be used in the conformance test suite, but should be only called in a test case.

Each test case is responsible for correct reaction on verdicts.
If a partial or preliminary FAIL is encountered, this should lead to a correct sequence:

not to issue send commands that are thought only in case of PASS
run the correct postambles if necessary for recovery
leave the test case in orderly fashion
(See chapter 6.1.2)

The next test case in suite has not to be disturbed!

Each test group, suite, subgroup and case (e.g. in CASE: and CASE;) contains two 'DOER' constructs at the begin and end: (see Chapter 12.4)

The DOERs may contain:

Print/Disk log control, depending on result
Identification frame sending
Host handshake

HOW TO CREATE A NEW TEST SUITE IN THE TEST SELECTION MENU:

1. Define the **test cases** and **steps**!
2. Create a tree structure of all test cases.
3. Attach your suites to the 'Test Selection' Menu.

7.1 Define TEST CASES / STEPS

STEP: **(--)**
Starts a test step definition.
Example: `STEP: STEP_name`

STEP; **(--)**
Ends a test step definition

CASE: **(--)**
Starts a test case definition.
Example: `CASE: CASE_name`

CASE; **(--)**
Ends a test case definition

-" **(-- string address)**
Compile string into dictionary.
String definition for the objects and test cases, groups, subgroups, and suites.
Example : -" comment " ,
The difference to the normal quote " is, that the literal buffer is not used
but the string is compiled into dictionary to stay accessible.

Note: Don't forget the comma following the long name string!

A test step may look like:

```
ON ATCM                                          ( switch indentation compiler on)
        -" This is a long name for STEP1 " ,
        STEP:    +STEP1                          ( test step start&name )
                    ! Send_msg1
                          ? Receive_msg2
                                Action1
                             ! Send_msg3
                                   ? Receive_msg4
                                           Action2
        STEP;                                    ( test step subtree end )
OFF ATCM                                         ( switch indentation compiler off )
```

A test case may look like:

```
ON ATCM                                          ( switch indentation compiler on)
        -" Test case for purpose ... " ,
```

```
        CASE:    CASE1                          ( test case start and name)
                 +STEP1
                          ?  Receive_msg1
                                 PASS
                          ?  Receive_msg2
                                 INCONC
                          ?  Receive_msg3
                                 FAIL
        CASE;                                   ( test case end )
OFF ATCM                                        ( switch indentation compiler off )
```

The called test step +STEP1 in the test case will be expanded, how desribed in chapter 6.1.1.

7.2 Create SIETEM tree structure

GROUP: (--)
Starts a test group definition.
Example: `GROUP:` GROUP_name

GROUP; (--)
Ends a test group definition

SUBGROUP: (--)
Starts a test subgroup definition.
Example: `SUBGROUP:` SUBGROUP_name

SUBGROUP; (--)
Ends a test subgroup definition

SUITE: (--)
Starts a test suite definition.
Example: `SUITE:` SUITE_name

SUITE; (--)
Ends a test suite definition

Example:

```
ON ATCM
     -" long name" ,
     SUBGROUP: SUBGROUP1
        CASE1
     SUBGROUP;
OFF ATCM

ON ATCM
     -" long name" ,
     GROUP: GROUP1
        SUBGROUP1
     GROUP;
OFF ATCM

ON ATCM
     -" long name" ,
     SUITE: SUITE1
        GROUP1
     SUITE;
OFF ATCM
```

7.3 Attach the TEST SUITE to the MENU SYSTEM

There are 10 slots for TEST SUITEs. To attach a test suite at position 2, type:

```
2  SET_SUITE  SUITE_name
```

SET_SUITE sets the named test suite definition into the selection menu ' TEST SELECTION' at the given slot. The 10 slots are numbered from 1 to 10.

SET_SUITE **(# of slot --)**
The slots are filled initially with EXIT. (1...10 SET_SUITE EXIT). The EXIT is an indication for the display not to show the line and stop.

AUTO_SET_SUITE **(--)**
Like SET_SUITE, but automatically creation of next slot number.

8 CASE SELECTION TOOL

Now, the structure for selecting and deselecting of test cases, subgroups, groups and suites (CGS) will be decsribed.

The final decision, if a CGS is run, is found with the word SELECTED?:

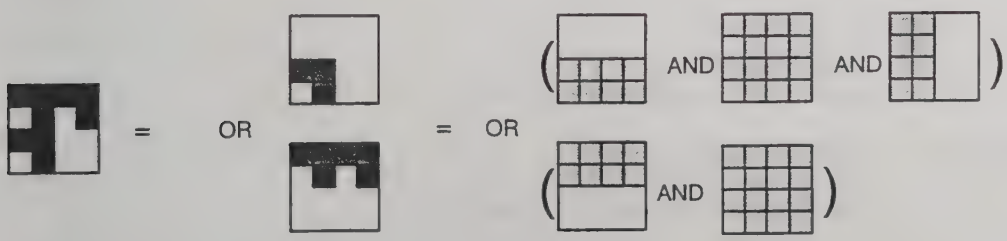
SELECTED? **(PFA of CGS -- FLAG)**
Is the CGS selected according to the 'SEL-BIT-TABLE'.
SELECTED? uses a truth table for the decision.

KARNAUGH TABLE for SELECTED?:

	NOT SEL?	SEL?	NOT SEL?	SEL?	
NOT RESTRICTED	1	1	1	1	NOT REGULATORY
RESTRICTED	0	1	0	1	
NOT RESTRICTED	1	1	0	0	REGULATORY
RESTRICTED	0	1	0		
	NOT LOCKED?		LOCKED?		

In textual form, the setting of '1' in this combination means:

SELECTED? = (REGULATORY AND (SEL? OR NOT RESTRICTED) AND NOT LOCKED)
OR
(NOT REGULATORY AND (SEL? OR NOT RESTRICTED))



Usually, 'SELECTED?' has to be defined with IF-CLAUSES, but using the KARNAUGH-table is a much shorter form!

To input this table there are two variables

RESTRICTED_CASES/GROUPS/SUITES

REGULATORY_CAMPAIGN

and a table definition:

```
0X01010101  VARIABLE SEL-BIT-TABLE
0X00010001 ,
0X01010000 ,
0X00010000 ,
```

Note: When e.g. a group is deselected, it does not run. That does not mean, that all the contained cases are deselected. When the deselected group is entered, it is still possible to run the selected test cases if expressively wanted.

SOME WORDS:

Here are some words to change select and lock bit of a CGS:

SELECT_CGSP **(PFA OF CGS --)**
set select bit to 1

SELECT_CGS **(--)**
set select bit to 1
Usage: `SELECT_CGS SUITE_name`

DESELECT_CGSP **(PFA OF CGS --)**
set select bit to 0

DESELECT_CGS **(--)**
set select bit to 0
Usage: `DESELECT_CGS SUITE_name`

LOCK_SELECTP **(PFA OF CGS --)**
set the lock bit to 1
Usage: `' SUITE_name LOCK_SELECTP`

LOCK_SELECT **(--)**
set the lock bit to 1
Usage: `LOCK_SELECT SUITE_name`

UNLOCK_SELECTP **(PFA OF CGS --)**
set the lock bit to 0
Usage: `' SUITE_name LOCK_SELECTP`

UNLOCK_SELECT **(--)**
set the lock bit to 0
Usage: `LOCK_SELECT SUITE_name`

SEL? **(PFA -- FLAG)**
is the cgs's select bit on ? PFA is the respective CGS's parameter field address, as may be obtained with '

LOCKED? **(PFA -- FLAG)**
is the CGS's lock bit on ?

CURRENT-CGS-PFA **(-- PFA)**
Gives the PFA of the currently running CGS.

CURR_CGS_SEL? **(-- FLAG)**
is the running case/group/suite selected ?

9 VERDICTS

At the end of each test step / case / subgroup / group / suite, the resulting verdict is transferred to the next higher structure, using the verdict rules from ISO 9646.
Then the name is displayed/recorded together with the verdict.

9.1 Verdict words

According to ISO 9646 there are final verdicts and preliminary verdicts.

Final verdicts:

PASS (--)
Give pass verdict, to trace

P (--)
Short form of PASS

The TCAP word **P** had to be redefined in SIETEM! Instead of **P** the word **PD** can be used. The word 'P' in SIETEM is the abbreviation of 'PASS'.

INCONC (--)
Inconclusive verdict

INC (--)
Short form of INCONC

FAIL (--)
Fail verdict

F (--)
Short form of FAIL

ERROR_VERDICT (--)
ERROR verdict in case of a system error

Preliminary verdicts:

(PASS)	**(INCONC)**	**(FAIL)**
(P)	**(I)**	**(F)**

Word used for constructs, depending on verdicts:

?PASS (-- flag)
Delivers true if verdict = PASS

VERDICT? **(pfa -- verdict number)**
Which verdict does a test case / group / suite have?
The verdict number means:
0 = NONE
1 = PASS
2 = INCONC
4 = FAIL
8 = ERROR of system

9.2 TEST RUN

After starting the test suite/group/subgroup or the test case on which the cursor is positioned with **RUN_SEL (F8)**, if selected in the menu, the text at the beginning of each test suite/test group/test case is marked blue on the screen followed by a detailed description of the started test part.
At the end of each test case the same presentation appears, followed by one of the possible verdicts PASS, FAIL or INCONC.

There is the following colour assignement:

PASS marked blue
FAIL marked red
INCONC marked yellow

At the end of a test group or a test suite a global verdict is added in the same way. In the test run, temporary verdicts appear in brackets with the same colour assignement.

10 TEST SUITE EXAMPLE

This example running on CCS#7 shows a test suite with test groups / subgroups containing test cases, which calls test steps and the used utility words:

Utilities:

```
47 CONSTANT WAIT_FOR_PAR_TIMER
48 CONSTANT Maintenance-TIMER
0  VARIABLE DeleteIMSI1.IID      ( Local Variable for Invoke ID )
0  VARIABLE DeleteIMSI1.IMSI     ( Local Variable for IMSI )
0  VARIABLE PAR_COUNTER


: WAIT_FOR_PARS  ( Number of parall. parts -- flag )
  1
  DOCASE
     CASE 1 1 ?TM_IPC
                     { PAR_COUNTER 1+
                           .
                           .
                           .
                     }
     CASE 2 2 ?TM_IPC
                     { PAR_COUNTER 2+
                           .
                           .
                           .
                     }
          .
          .
          .
  ENDCASE

    .
    .
    .
;
```

PCO – Point of Control and Observation (See ISO/IEC–DIS 9464–3):

```
: HLR_MAP-D ( -- flag ) ( RECEIVE ) ( EDIT_SCCP_TCAP had to be )
            ( -- )      ( SEND )    ( done before )
  IF?       ( RECEIVING ? )
    ?LINK 1 =                  ( CHECK IF LINK CORRECT - HLR )
    IF
      0X307E ?DUP              ( SPC for HLR )
      IF                       ( CHECK DPC IF <> 0 )
        ?MSU_TO_DPC
        FALSE->                ( See Chapter 12.1 )
      THEN
          .
          .
      #GSM_SCCP #SCCP_UDT ?RX_SU      ( IS IT SCCP UNIT DATA ? )
          FALSE->
      SCCP_CLD_SSN P@                 ( Check if called SSN = HLR  )
          FALSE->
            6 =                       ( #SSN for HLR )
          FALSE->
          .
          .
          .
       SCCP_DATA_DATA P@     ( Check if DATA parameter is present )
          FALSE-> DROP
          .
       ( TCAP emulation will be called )
          .
          .
    ELSE
        V." Message for other LINK received in PCO: HLR_MAP-D" VCR
          .
          .
          .
    ENDIF
  ELSE    ( SENDING )
          .
    ( TCAP emulation will be called )
          .
          .
    SEND_SCCP_TCAP
  ENDIF
;
```

This definition of a PCO explicitly handles I/O and lower layer emulation. It checks received messages if they fit in this PCO. When messages are sent via this PCO lower layer support is automatically included.

Constraints (See ISO/IEC–DIS 9464–3):

```
: DISUP_REL                    ( flag -- flag )
  IF
    #D1ISUP #DI_REL ?RX_SU
        FALSE->
          .
          .
          .
    1
  ELSE
    0
  ENDIF
;

: TcResultLInd[DID2,I]         ( flag -- flag )
  IF
     #TCAP #MSG_END ?TCAP_SU
           FALSE->
          .
          .
          .
     INVOKE_ID  T@
           FALSE->
        DeleteIMSI1.IID @ =
           FALSE->
     RRL_COMPONENT T@
           FALSE-> DROP
     1
  ELSE
     0
  ENDIF
;
```

PREAMBLE:

```
ON ATCM

STEP: +PREAMBLE
          .
          .
       ( Initialize Links )
       ( Load Pool messages )
          .
          .

STEP;

OFF ATCM
```

POSTAMBLE:

```
ON ATCM

STEP: +POSTAMBLE
          .
          .
       ( Release all pending dialogues and disconnect emulations )
          .
          .
          .
       !SHUTDOWN
          .
STEP;

OFF ATCM
```

Default behaviour:

```
ON ATCM

DEFAULT: TESTEND
       ? Maintenance-TIMER ?TM_TIMER
          ULTIMATE_FAIL
          (FAIL)
          +POSTAMBLE
            ? ?SHUTDOWN
              EXIT->
          .
          .

       ? MSC1_ISUP DISUP_REL
          REMAIN_DEFAULT
          .
          .
          .

       ? BSS1_BSSAP
          ULTIMATE_FAIL
          (FAIL)
          +POSTAMBLE
            ? ?SHUTDOWN
              EXIT->
          .
          .
          .

       ? ?SHUTDOWN
          (FAIL)
          EXIT->
DEFAULT;

DEFAULT= TESTEND

OFF ATCM
```

Test step:

```
ON ATCM

STEP: +DeleteIMSI1
      " TcB-INV_CancLoc" EDIT_SCCP_TCAP        ( GET MESSAGE FROM POOL)
         .
         .
      ! HLR_MAP-D
         .
         .
         .

        Maintenance-TIMER 100 START_TM_TIMER
            ?  HLR_MAP-D  TcResultLInd[DID2,I]
               Maintenance-TIMER STOP_TM_TIMER
               (PASS)
                 .
                 .
                 .
            ?  HLR_MAP-D
               Maintenance-TIMER STOP_TM_TIMER
              (INCONC)
STEP;

OFF ATCM
```

Test cases:

```
ON ATCM

-" LUIR/CT/NormalUpdatewithIMSI" ,
CASE: 2.1.0.1
      +PREAMBLE
        " 262011500000001"  DeleteIMSI1.IMSI !STRING
        +DeleteIMSI1
         .
         .
         .
CASE;

OFF ATCM
```

```
ON ATCM

STEP: 2.1.1.2.1.2.PAR1
          .
          .
          .
       ! 1 0 TM_IPC_SEND            ( Send coordination message CM to
                                      coordinatin point CP )
STEP;


STEP: 2.1.1.2.1.2.PAR2
          .
          .
          .
      ! 2 0 TM_IPC_SEND             ( Send coordination message CM to
                                      coordinatin point CP )

STEP;


-" LUIR/NB/AUT/DVSP/4AUTHsets" ,
CASE: 2.1.1.2.1.2
     +PREAMBLE
          imsi1  DeleteIMSI1.IMSI !STRING
        +DeleteIMSI1
            .
            .
            .
                    1 ' 2.1.1.2.1.2.PAR1 ->TM( Starts two parallel parts )
                    2 ' 2.1.1.2.1.2.PAR2 ->TM  ( See Chapter 13.2 )
                    WAIT_FOR_PAR_TIMER 300 START_TM_TIMER
                      ? 2 WAIT_FOR_PARS       ( Wait for end message of both
                                                parallel parts )
                           WAIT_FOR_PAR_TIMER STOP_TM_TIMER
                           (PASS)
                       ? WAIT_FOR_PARS_TIMER ?TM_TIMER
                           (FAIL)
                      .
                      .
                      .
                      .

CASE;

OFF ATCM
```

Test suite:

```
ON ATCM

-" LUIR/Capability Tests (CT)" ,
GROUP: 2.1.0
       2.1.0.1
GROUP;

-" LUIR/NB/Initial Message (IM)" ,
GROUP: 2.1.1.1
        .
        .
        .
GROUP;

-" LUIR/NB/AUT/DataVarSendP(DVSP)" ,
SUBGROUP: 2.1.1.2.1
          2.1.1.2.1.2
        .
        .
        .
SUBGROUP;

-" LUIR/NB/Authentication (AUT)" ,
GROUP: 2.1.1.2
       2.1.1.2.1
GROUP;

-" LUIR/NB/Identification (ID)" ,
GROUP: 2.1.1.4
        .
        .
        .
GROUP;

-" LUIR/Normal Behaviour (NB)" ,
GROUP: 2.1.1
       2.1.1.1
       2.1.1.2
       2.1.1.4
GROUP;

-" LU/Initial Registration (LUIR)" ,
GROUP: 2.1
       2.1.0
       2.1.1
       .
       .
       .
GROUP;
```

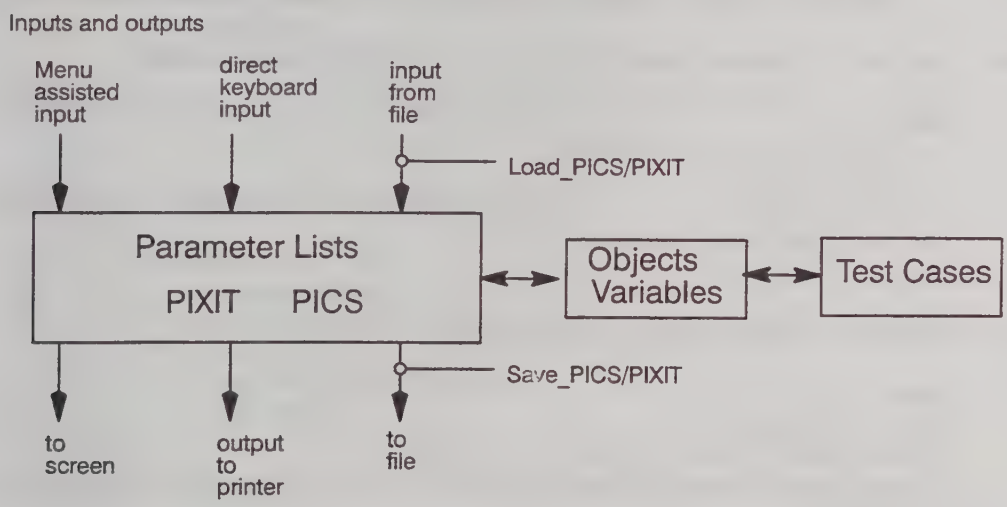
```
-" LU/IMSI known in VLR (LUIK)" ,
GROUP: 2.2
       .
       .
       .
GROUP;

-" LU/IMSI known in PVLR (LUPV)" ,
GROUP: 2.3
       .
       .
       .
GROUP;

-" Location Update" ,
SUITE: LU
        2.1
        2.2
        2.3
SUITE;

AUTO_SET_SUITE LU

OFF ATCM
```

11 PICS/PIXIT PARAMETERS

PICS / PIXIT Parameters are objects used like variables, they will be needed to initialize the test cases surrounding.
The advantage of them is the possibility to change the values via menu system!

The objects contain some methods (functions to be performed on object data) and some data. There are several classes: enumerated variables, decimal number variables, hex strings, data frames etc.

The environment looks as follows:



HOW TO CREATE PICS/PIXIT PARAMETERS:

1. Define the **objects** with the default values!

2. Create a new menu entry (**PICS–List**)!

11.1 Define PICS/PIXIT–objects

ENUM_PICS Definition for Enumerated PICS/PIXIT Object.

Usage:
1.) Number of parameters (Constants)
2.) Default_value
3.) **ENUM_PICS**
4.) Name of PICS/PIXIT parameter
5.) List of Constants

Example:
```
2  ON  ENUM_PICS  RESTRICTED_CASES/GROUPS/SUITES   ON  OFF
```

ENUM_PARAM Definition for enumerated PICS/PIXIT Object to use an already defined Variable.

Usage:
1.) Number of parameters (Constants)
2.) Name of Variable
3.) **ENUM_PARAM**
4.) Name of PICS/PIXIT parameter
5.) List of Constants

DECIMAL_PICS Definition for Decimal Number PICS/PIXIT Object

Usage:
1.) –" Prompt String"
2.) Lower_bound (positive 31bit number 0.. 2x10 ** 9)
3.) Upper_bound (positive 31bit number 0.. 2x10 ** 9)
4.) Default_value
5.) **DECIMAL_PICS**
6.) Name of PICS/PIXIT parameter

Example:
```
–" INPUT TIME IN TENTH OF SECONDS " 0 10000
0 DECIMAL_PICS WAIT_TIME_AFTER_CASE
```

DECIMAL_PARAM Definition for Decimal Number PICS/PIXIT Object to use an already defined variable.

Usage:
1.) –" Prompt String"
2.) Lower_bound (positive 31bit number 0.. 2x10 ** 9)
3.) Upper_bound (positive 31bit number 0.. 2x10 ** 9)
4.) **DECIMAL_PARAM**
5.) Name of PICS/PIXIT parameter

HEXSTRING_PICS Definition for Hex String Pics/Pixit Object.

Usage:
1.) –" Prompt String"
2.) Maximal length (Maximal Size for input strings: 80 characters.)
3.) **HEXSTRING_PICS**
4.) Name of PICS/PIXIT parameter
5.) Initial HEX string"

Example:

```
–" INPUT HEX STRING. 3 OCTETS: 1. A–H, 2. Timeslot, 3. I–K"
3 HEXSTRING_PICS circ1 010203"
```

Note: Do not insert more spaces or returns between 'HEXSTRING_PICS' and the last double quote!

HEXSTRING_PICS outputs its string on SAVEPARAM (save parameter list in file) with a 0X prefix.

STRING_PICS Definition for ASCII String Pics/Pixit Object.

Usage:
1.) –" Prompt String"
2.) Maximal length (Maximal Size for input strings: 128 characters.)
3.) **STRING_PICS**
4.) Name of PICS/PIXIT parameter
5.) Initial String"

The string objects allow empty strings to be entered. This results in strings of zero length .

The string objects should not contain spaces (0X20), otherwise a loading of the parameter list (LOADPARAM: see Chapter 3.1) will fail.

F1_PICS Definition for enumerated Function PICS/PIXIT Object, executed on P_EXECUTE and invocation (see chapter 10.2).

Usage:
1.) Number of parameters (Functions)
2.) **F1_PICS**
3.) Name of PICS/PIXIT parameter
4.) Default Function
5.) List of Functions

F2_PICS Definition for enumerated Function PICS/PIXIT Object, executed like F1_PICS and also on P_CHANGE (see chapter 10.2).

Usage:
1.) Number of parameters (Functions)
2.) **F2_PICS**
3.) Name of PICS/PIXIT parameter
4.) Default Function
5.) List of Functions

11.2 Create a new MENU SYSTEM entry

To create a new menu entry in the PARAMETER_LISTS MENU you have to write the PICS/PIXIT objects (see chapter 11.1) in a PICS–List:

PICSL: **(– –)**
Starts a PICS/PIXIT–Parameter list definition.
Example: `PICSL: PICS_name`

PICSL; **(– –)**
Ends a PICS/Parameter list definition

–" **(– – string address)**
Compile string into dictionary.
String definition for the objects and test cases, groups, subgroups, and suites.
Example : `–" STRING " COUNT TYPE`
The difference to the normal quote " is, that the literal buffer is not used
but the string is compiled into dictionary to stay accessible.

V.C" Comment string definition for parameter lists
Example: `V.C" ( this is a string to be displayed ) " VCR`
Caution: 1. Immediately after **V.C"** a space has to be left free.
2. The string should be in brackets. This avoids compiling of the string while LOADPARAM is running. The open bracket should find a space at its left and right hand side.

Now the PICS–List to the MENU SYSTEM is to be attached:

There are 20 slots for PICS / PIXIT Parameter lists. To attach a list at position 2, type:

```
2  SET_PARAM  PICS_name
```

SET_PARAM sets the named parameter list definition into the selection menu 'PARAMETER LISTS' at the given slot. The 20 slots are numbered from 1 to 20.

SET_PARAM **(# of slot – –)**
The slots are filled initially with EXIT. (1...20 SET_PARAM EXIT). The EXIT is an indication for the display not to show the line and stop.

AUTO_SET_PARAM **(– –)**
Like SET_PARAM, but automatically creation of next slot number.

11.3 Example

```
1 CONSTANT Constant_1
2 CONSTANT Constant_2


2  Constant_1  ENUM_PICS  PICS-Variable_1  Constant_1 Constant_2

-" INPUT VALUE: 0-10 " 0 10
0 DECIMAL_PICS PICS-Variable_2

-" INPUT HEX STRING. 3 OCTETS:"
3 HEXSTRING_PICS PICS-Variable_3 010203"


-" Long name" ,
PICSL: PICS_name
    V.C" ( Comment ) " VCR
    PICS-Variable_1    VCR
    V.C"              " VCR
    PICS-Variable_2    VCR
    PICS-Variable_3    VCR
PICSL;

AUTO_SET_PARAM PICS_name
```

Note: Each PICS / PIXIT or string has to be separated with VCR!
Never use two VCRs directly in sequence!
Empty lines are given with `V.C" " VCR`.
The long name will appear in the parameter lists selection menu.

11.4 Words used by function: SAVEPARAM

Now follow some definitions for the SAVEPARAM and LOADPARAM functions.(See Chapter 3.1)
The respective assignment symbols are used when the current contents of objects in a parameter list is written to disk or to screen. When parameter files are loaded back with LOADPARAM, the assignment symbols move the indicated contents into the objects.

:=	Value assignment for longs (ENUM_PICS, ENUM_PARAM, DECIMAL_PICS, DECIMAL_PARAM)
:$=	Value assignment for strings (STRING_PICS, HEXSTRING_PICS)
:*=	Value assignment for functions (F1_PICS, F2_PICS).

11.5 ' METHOD '

'METHOD' is a utility to change the behaviour of PICS/PIXIT parameters.

METHOD Constant. Holds the actual method for parameter objects

P_ACCESS (--)
In this case the objects behave like ordinary FORTH variables. They have an address, pointing to the data, on stack, and thus allow for reading (and writing) the values.
METHOD value = 0
Standard value for METHOD while executing test cases!!

P_SCREEN (--)
As the objects use the vectored output routines (V. routines), the output may be directed to screen, printer, RAM etc.
For disk output, first a file should be opened, then a Disk Track Buffer should be allocated and then the V.RAM directs the output to the buffer, where a VCR command may check each line if the buffer has to be written to disk.
METHOD value = 1
Example:
```
P_SCREEN
V.SCREEN
VERDICT-LOG
```
Output:
```
VERDICT-LOG := YES
```

P_CHANGE (--)
This method is usable when the PICS / PIXIT list is displayed on screen and the user wants assistance to adjust the values. When an object is called with this method, the action depends on the object class. (Used for F2_PICS)
METHOD value = 3

P_EXECUTE (--)
Set Execute Method (before using F1_PICS or F2_PICS)
METHOD value = 4

P_DEBUG **(--)**

Debug output to control the syntax of a PICS/PIXIT parameter.
METHOD value = 99

Example:
```
P_DEBUG
        VERDICT-LOG
```

Output:

```
CONTENTS OF VERDICT-LOG
  C94FDE 00 c9 59 4E 00 00 00 02 00 C8 8D D2 00 C8 8D C2 ..YN.............
  C94FEE 00 8A 45 4E 55 4D 5F 4D 45 4E 55 34 00 C9 4F C6 ..ENUM_MENU4 ..O .
VARIABLE= DO-VLOG* VALUE = 00000001 YES
NUMBER OF PARAMS = 2
00C88DD2 YES
00C88DC2 NO
```

12 OTHER WORDS

CASE_DISPLAY **(ON | OFF --)**
If `ON CASE_DISPLAY`, at the top right hand corner of the screen the **actual running test case** instead of the time will be shown.

BETWEEN? **(number / low / high -- flag)**
Checks if 'number' lies between (inclusive) low and high.

12.1 Debugging words

DEBUG Variable: Value: 0 – No debugging
Value: > 0 – The higher the value so much the more debug outputs will be given.

This words have to been called within test cases / groups ... :

T **(--)**
Trace transmit command.

T" **(--)**
Handling of command trace strings.
Example:

```
SABM T" UA OR DM [F=1] "
```

Output trace:

```
SENT : SABM , EXPECT: UA OR DM [F=1]
```

.RX **(--)**
Display last received event

.RX– **(--)**
Display last received event without VCR

STA **(--)**
Outputs current TTCN stack.

The following word is a utility for writing time–optimized receive constraints and PCO's (see chapter 10). On the other hand it gives debugging information, if a receiving condition was not met.

FALSE–> **(0 -- 0 EXIT)**
(1 --)
Exist for Event Recognition.
Use ONLY in FORTH definitions (Constraints, ...) and NEVER in test steps and test cases. (See chapter 10)
USAGE:
```
: CONSTRAINT_name
  #GSM_SCCP #RLCS ?RX_SU
     FALSE->
;
```

F–>D **(--)**
DOER word to change the length of debugging output of FALSE–>

Example:

```
MAKE F->D DROP             ;
MAKE F->D DEBUGS_FALSE-> ;
MAKE F->D DEBUGM_FALSE-> ;
```

No debugging output of FALSE–>
(Initially usage of FALSE–>)
Short output of FALSE–>
Longer output of FALSE–>

12.1.1 ' SPY '

There is a mechanism to observe the current stack behaviour of Test steps / cases / groups, Defaults and Constraints. The contents of the stack will be displayed before and after running.

SPY_ON_TTCN **(--)**
Secures a TTCN definition with the 'SPY' mechanism.

SPY_OFF_TTCN **(--)**
Switches the 'SPY' mechanism off for a TTCN definition.

Example:

```
SPY_ON_TTCN testcase_name
SPY_ON_TTCN constraint_name
.
.
1 10 TRACE_WINDOW TRACE
```

12.2 Vectored output words

These routines work similarly to the normal FORTH output routines, but the output may be directed to screen, printer etc.

V. **(number --)**
Output a number (in decimal)

V." **(--)**
Output an inline string

V.C" **(--)**
Output string in parameter list

V.ID. **(nfa --)**
Output a name of a function

VCR **(--)**
Output a carriage return (new line)

V.H **(number --)**
Output a number in Hex

V.HB **(number --)**
Output hex byte .HB

V.HH **(number --)**
Output hex nibble .HH

V.TIME **(time --)**
Output time as hh:min:ss

V.TYPE **(string pointer/count --)**
Output ascii string of length 'count'

V.TYPH **(string pointer/count --)**
Output hex string of length 'count'

V.DATE **(--)**
Output date as yy/mm/dd

V.R **(number/field length --)**
Output number in given field .R

V.SPACES **(count--)**
Output up to 77 spaces

V.DUMP **(Adr / lgth --)**
Output contents in HEX

12.2.1 Select the output channel

Following words select the output channel of the " **V.** " (vectored) words:

V.SELECTED Variable, indicates current output channel set by following words.

V.SCREEN (--)
to screen only.For debugging and hints .
V.SELECTED value = 1

V.PRINT (--)
to printer only.Caution: if used in tests in receive loops,a slow printe may cause a timeout,when a full print buffer accepts no further print commands.
V.SELECTED value = 2

V.SCREEN+PRINT
(--)
to screen and printer.
V.SELECTED value = 3

V.TRACE (--)
direct to trace: screen,capture buffer,disk.
V.SELECTED value = 4

V.TRACE+PRINT
(--)
to trace and printer .
V.SELECTED value = 6

V.CAPTURE (--)
to capture buffer only.For debugging and hints .
V.SELECTED value = 8

V.RAM (--)
direct output to RAM.
V.SELECTED value = 16

V.RAMBUF (--)
A 170 byte RAM buffer for V.RAM.

12.3 TREE

TREE is a utility to create a list of all test cases in a suite. This list shows the short and long names of test cases , subgroups and groups in a "TREE–structure".

INDENTING Variable, indicating the indentation level. Default=5

Usage:

3. Select Output Channel:
   ```
   V.SCREEN ,
   V.RAM     or    ( For output to Disk )
   V.PRINT
   ```

4. Call Tree with the Structure Root Name as Parameter:
   ```
   TREE SUITE_name
   ```

5. If V.Ram was selected, type:
   ```
   " filename" WTD
   ```
   when asked to write the ASCII data to disk.

Example of output:

```
LU :  Location Update
2.1  LU/Initial Registration (LUIR)
     2.1.0  LUIR/Capability Tests (CT)
        . 2.1.0.1  LUIR/CT/Simple
     2.1.1  LUIR/Normal Behaviour (NB)
          2.1.1.1  LUIR/NB/Initial Message (IM)
               2.1.1.1.1  LUIR/NB/IM/DataVar-CLI3(DVCL)
                    2.1.1.1.1.1  LUIR/NB/IM/DVCL3/minLAC
                    2.1.1.1.1.2  LUIR/NB/IM/DVCL3/minLACmidCI


.
.
.
.
               2.3.2.4.4  LUPV/AB/UHLR/AbortBSS
               2.3.2.4.5  LUPV/AB/UHLR/Unexp.Data
               2.3.2.4.6  LUPV/AB/UHLR/Timeout
          2.3.2.5  LUPV/AB/Ciphering (CI)
               2.3.2.5.2  LUPV/AB/CI/AbortHLR_CiphFail
          2.3.2.6  LUPV/AB/TMSIreallocation (TMS)
               2.3.2.6.1  LUPV/AB/TMS/Timeout
               2.3.2.6.2  LUPV/AB/ClearRqst
               2.3.2.6.3  LUPV/AB/DiscInd
     2.3.3  LUPV/Complex Behaviour (CB)
          2.3.3.1  LUPV/CB/UnknSubsc.PLVR-HLR
          2.3.3.2  LUPV/CB/UnknTMSIinVLR_T3260
```

12.4 'DOER' words

There are some DOER words reserved:

Note: Do NOT delete the contents of a DOER word!
Do ONLY add a function to the DOER word!

To change the contents of a DOER word:

1. Look at the old function of it:

```
DECOMP DOER_CASE1:
```

Output:

```
DOER DOER_CASE1: function_old EXIT
```

2. Define a new word with the old AND new contents in it:

```
: NEW_WORD
  function_old
  functions_new1
  functions_new2
;
```

3. Change the function of it:

```
MAKE DOER_CASE1: NEW_WORD ;
```

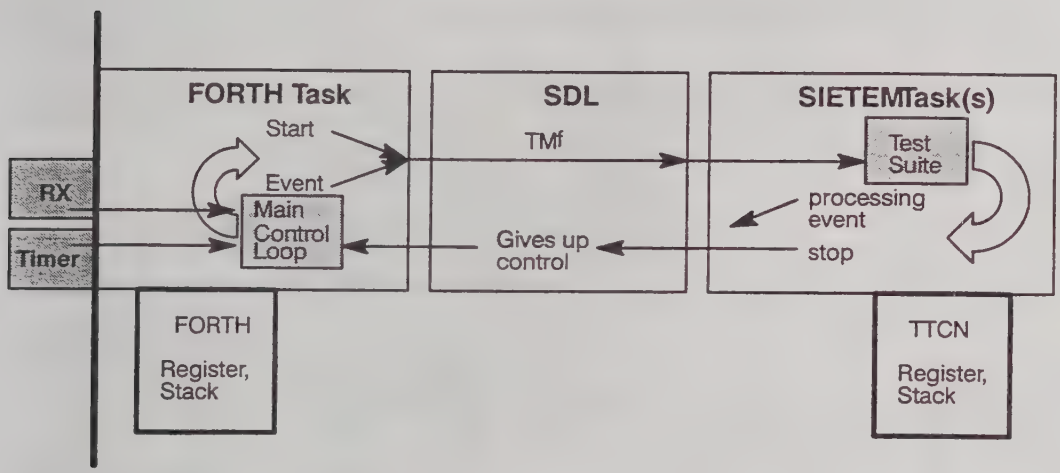
<DSP_VERS>	DOER word for the Software versions.*
DOER_STEP;	DOER words in STEP;
DOER_CASE1: DOER_CASE2:	DOER words in CASE:
DOER_CASE1; DOER_CASE2;	DOER words in CASE;
DOER_SUBGROUP1: DOER_SUBGROUP2:	DOER word in SUBGROUP:
DOER_SUBGROUP1; DOER_SUBGROUP2;	DOER word in SUBGROUP:
DOER_GROUP1: DOER_GROUP2:	DOER word in GROUP:
DOER_GROUP1; DOER_GROUP2;	DOER word in GROUP:
DOER_SUITE1: DOER_SUITE2:	DOER word in SUITE:
DOER_SUITE1; DOER_SUITE2;	DOER word in SUITE:
DOER_PASS	DOER word in PASS
DOER_FAIL	DOER word in FAIL

* Only applicable on CCS#7 with SSS–TE

DOER_ERRORV		DOER word in ERROR_VERDICT
DOER_INCONC		DOER word in INCONC
DOER_RUN_SEL		DOER word in RUN_SEL of the case selection menu
START_TEST_DOER	**END_TEST_DOER**	This doer words are called at the start / end of the test
START_CASE_SELECT		This doer word is called at the start of the case select menu
LEAVE_PICS_SELECT		This doer word is called when the PICS/PIXIT parameter menu is left. It is intended to use for calling the PICS/PIXIT dependent CASE SELECTION TABLE

13 SDL testmanager and SIETEM

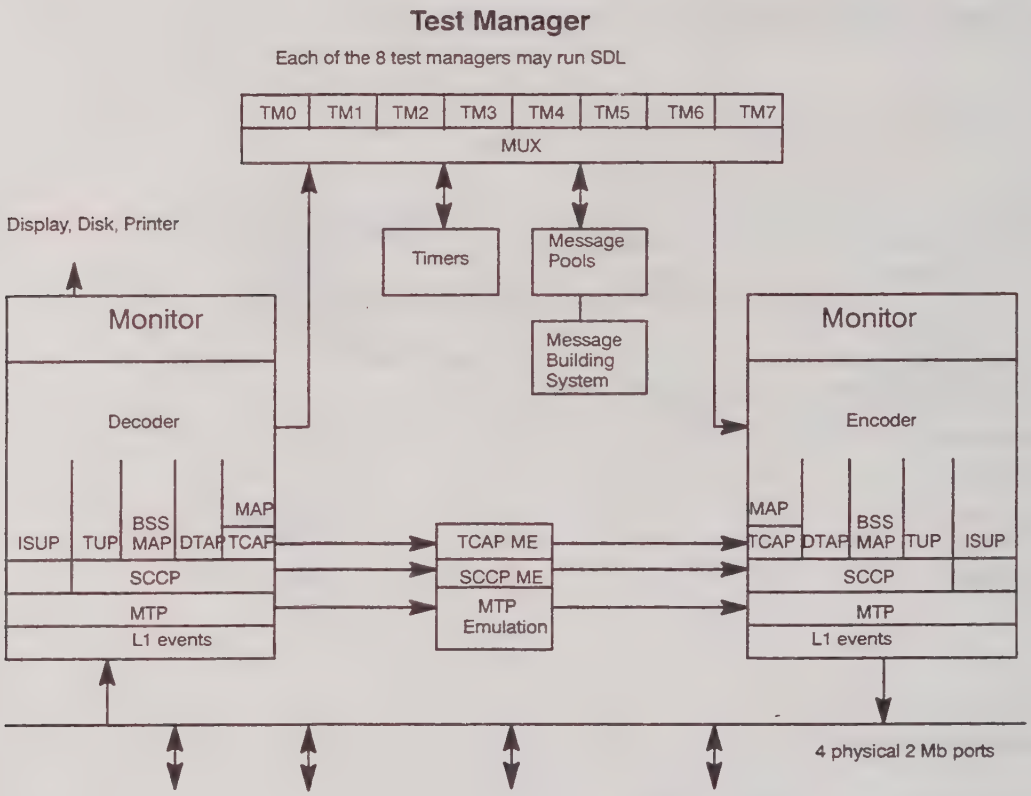
CONNECTION OF SIETEM TO THE CURRENT K1197 (SDL) TEST MANAGER:



Example for SIETEM called by SDL:

```
0 STATE{    OTHER_EVENT     ACTION{     TM{     }ACTION     }STATE
```

13.1 Overview



13.2 Enhancements for SIETEM

There are some enhancements using the SDL test manager for SIETEM:

–>TM **(No. of TM / PFA of test ––)**
Starts the test step / case ....

TM= **(No. of TM ––)**
Starts the following test step / case ....

TM{ **(––)**
Call the SIETEM test manager.
SYSTEM WORD!

13.3 Source files on binary diskette

The following files you find as source code on you binary disk.

Usually, those files are already compiled and saved in the binary:

TMTM.B	at SS No.7 application
TMTM.B1, ... TMTM.B8	at ISDN application.

They are a testmanager binarys which work as an interface between events (Frames, Timeouts, etc.) and the running test written in TTCN.

When executing the first test the file `TMTM.Bx` (x is the CPU–number 1,3,7 or 8) or `TMTM.B` (for MAIN) will be loaded or you can load it manually:

e.g.: `" TMTM.B" EXEC_TS.`

This file must be available on any disk or partition when starting the first test.

If those files are **already not loaded** and there is no file TMTM.Bx you can compile and save them with:

At SS No.7 for example:

```
" M-TMTM" EXECF
FORGET LDTEST
" TMTM.B" SAVETS
```

FILE: M–TMTM

```
( M-TMTM                Multi-testmanager is ON        )
( TT_TMTM               Loaded by M-TMTM               )
( S-TMTM                Multi-testmanager is OFF       )
( TMTM.B                Loaded by SIETEM               )

#IFDEF LDTEST
    FORGET LDTEST
#ENDIF
: LDTEST
    GLOB_TCLR
    TM0 " TT_TMTM" EXECF
    TM1 " TT_TMTM" EXECF
    TM2 " TT_TMTM" EXECF
    TM3 " TT_TMTM" EXECF
    TM4 " TT_TMTM" EXECF
    TM5 " TT_TMTM" EXECF
    TM6 " TT_TMTM" EXECF
    TM7 " TT_TMTM" EXECF
    GLOB_TM_RUN
    TM0
;

#IFDEF CCS7-K1197#
  MUX_ON
#ELSE
  L3_MUX_ON
#ENDIF
LDTEST
```

FILE: TT_TMTM

```
( M-TMTM          Multi-testmanager is ON    )
( TT_TMTM         Loaded by M-TMTM           )
( S-TMTM          Multi-testmanager is OFF   )
( TMTM.B          Loaded by SIETEM           )

TCLR

#IFDEF CCS7-K1197#

0 STATE_INIT{
        1 KEY-FLAG* !           ( Receive start via FK8 )
        #UNUSED =CN_LINK        ( Connnection for link 1 to 4 )
        #UNUSED =CN_MSPC        ( Connnection for My SPC )
        #UNUSED =CN_YSPC        ( Connnection for Your SPC )
}STATE_INIT
0 STATE{
        OTHER_EVENT ACTION{
               EVENT_TYPE
                DOCASE
                    CASE #EV_FRAME
                  ORCASE #EV_LEVEL1
                  ORCASE #EV_LEVEL2
                  ORCASE #EV_TEST
                  ORCASE #EV_TM_IPC
                  ORCASE #EV_TM_TIME_OUT
                  ORCASE #EV_COMMAND
                  ORCASE #EV_EMULATION      { TM{ }
                    CASE #EV_TIME_OUT       { TIMER-NUMBER @ 51 <
                                              IF TM{ ENDIF  ( if user timer )
                                              }
                    CASE #EV_FUNCTION_KEY   { KEY-NUMBER @ UF6 =
                                              IF
                                                1 SUSPEND-FLAG !
                                              ELSE
                                                KEY-FLAG* @ IF TM{ ENDIF
                                              ENDIF
                                              }
                    CASE #EV_STRING_ENTRY   { STRING-FLAG* @ IF TM{ ENDIF
                                              }
                    CASE DUP            { }
                ENDCASE
        }ACTION
}STATE
```

```
#ELSE

0 STATE_INIT{
    0 KEY-FLAG* L!
    0 STRING-FLAG* L!
}STATE_INIT
0 STATE{
    DL-DATA#CONF ?L2_SERVICE
         ACTION{ }ACTION                    ( ignore acknowledging RRs )

    UF6 ?KEY
    ACTION{
             1 SUSPEND-FLAG L!
    }ACTION
    OTHER_EVENT
    ACTION{
        EVENT-TYPE @
        DOCASE
            CASE STRING-ENTRY# {    STRING-FLAG* @ IF TM{ ENDIF    }
            CASE FUNCTION-KEY# {    KEY-FLAG* @    IF TM{ ENDIF    }
            CASE DUP           {    TM{                            }
        ENDCASE
    }ACTION
}STATE

#ENDIF
```

13.4 Example

This example makes use of the SIETEM enhancements:

Four SDL–Test Managers are used to call parallel SIETEM test cases.

Testmanager: TM0

```
SDL: TT_TMTM

0 STATE{
 TM{
}STATE


TTCN: T0

CASE: T0

? UF1 ?KEY
 2 1 TM_IPC_SEND

? 1 1 ?TM_IPC

? 1 2 ?TM_IPC

? UF2 ?KEY
 1 1 TM_IPC_SEND
 1 2 TM_IPC_SEND

CASE;
```

Testmanager: TM1

```
SDL: TT_TMTM

0 STATE{
 TM{
}STATE


TTCN: T1

STEP: +PRE1

STEP;

CASE: T1
 +PRE1
 ? 2 0 ?TM_IPC

  ? #GISUP

 ? 1 ANY#TM

CASE;
```

Testmanager: TM2

```
SDL: TT_TMTM

0 STATE{
 TM{
}STATE


TTCN: T2

CASE: T2
 ? #GISUP

 ? 1 ANY#TM

CASE;
```

Testmanager: TM3

```
SDL: TT_TMTM

0 STATE{
 TM{
}STATE
```

Compile this testscenario with:

```
" TTL.F" EXEC
```

Start this testscenario manually with:

```
" M-TMTM" EXECF
0 TM= TEST1
```

Start this testscenario by SIETEM menu system:

select TEST1
f8 RUN_SEL

Save this testscenario as a binary:

```
" TEST1.B" SAVESU
" TMTM.B" SAVET
```

Load this testscenario by SIETEM menu system: See chapter 2.1.

Load SIETEM (as Binary)
Load executable Testsuite
f8 RUN_SEL

Source code of those files:

FILE: TTL.F

```
MAKE FILES
    0
    " TTG.F"
    " TT0.F"
    " TT1.F"
    " TT2.F"
;

FILES LOAD_FILES
```

FILE: TT0.F

```
0 VARIABLE tn-run


ON ATCM

STEP: T0
    ? UF1 ?KEY
        V." T0: Mail to TM1" VCR
        2 1 TM_IPC_SEND
        REMAIN

    ? 1 1 ?TM_IPC
        V." TM1 finished" VCR
        -1 tn-run +!
        tn-run @ IF REMAIN ENDIF

    ? 1 2 ?TM_IPC
        V." TM2 finished" VCR
        -1 tn-run +!
        tn-run @ IF REMAIN ENDIF

    ? UF2 ?KEY
        V." T0: Shut down all TMs" VCR
        1 1 TM_IPC_SEND
        1 2 TM_IPC_SEND
STEP;

OFF ATCM
```

FILE: TT1.F

```
ON ATCM

STEP: +PRE1
    #SPAL3 DUP SD ML START_LINK                ( Start CCS#7 layer 2
                                                emulation on port 1 )
    #SPBL3 DUP SD ML SET_EMERGENCY START_LINK( Start it on port 2 )
    " TEST.P" LOAD_POOL                        ( All messages to be sent must
                                                be included in 'TEST.P' )
    1 KEY-FLAG* L!                             ( inform TMTM to allow
                                                 keyboard input to the test
                                                 cases )
STEP;

OFF ATCM

ON ATCM

STEP: T1
    +PRE1
    ? 2 0 ?TM_IPC
        V." T1: ISUP-Call" VCR
        " IAM" SEND_MSG

        ? #GISUP #DI_ACM ?RX_SU
            V." T1: Adress complete" VCR

          ? #GISUP #DI_ANS ?RX_SU
            V." T1: Connection" VCR

            ? 2 0 ?TM_IPC
                V." T1: Release" VCR
                " REL" SEND_MSG
                " RLSD" SEND_MSG

                ? #GISUP #DI_RLC ?RX_SU
                  V." T1: Release complete" VCR
                  1 0 TM_IPC_SEND

                ? 1 ANY#TM ?TM_IPC
                    V." T1: Mail received. Shutdown T1" VCR

            ? 1 ANY#TM ?TM_IPC
                V." T1: Mail received. Shutdown T1" VCR

        ? #GISUP #DI_UBM ?RX_SU
            V." T1: No connection possible" VCR

        ? 1 ANY#TM ?TM_IPC
            V." T1: Mail received. Shutdown T1" VCR

    ? 1 ANY#TM ?TM_IPC
        V." T1: Mail received. Shutdown T1" VCR
STEP;

OFF ATCM
```

FILE: TT2.F

```
ON ATCM

STEP: T2
    ? #GISUP #DI_IAM ?RX_SU
        " ACM" SEND_MSG
        " ANS" SEND_MSG

        ? #GISUP #DI_REL ?RX_SU

            ? #GISUP #DI_RLSD ?RX_SU
                " RLC" SEND_MSG
                  1 0 TM_IPC_SEND

            ? 1 ANY#TM ?TM_IPC
                V." T2: Mail received. Shutdown T2" VCR

        ? 1 ANY#TM ?TM_IPC
            V." T2: Mail received. Shutdown T2" VCR

    ? 1 ANY#TM ?TM_IPC
        V." T2: Mail received. Shutdown T2" VCR
STEP;

OFF ATCM
```

FILE: TTG.F

```
ON ATCM

-" ISUP-CALL" ,
CASE: TEST1
    2 tn-run L!
    1 ' T1 ->TM             ( Start case T1 on TM1 )
    2 ' T2 ->TM             ( Start case T2 on TM2 )
    T0

CASE;

-" LOADER" ,
GROUP: GROUP-1
    TEST1
GROUP;

-" SUITY" ,
SUITE: ALL
    GROUP-1
SUITE;

AUTO_SET_SUITE ALL

OFF ATCM
```

INDEX

Symbols

.RX, 12–1
.RX–, 12–1
!, 6–1
!SHUTDOWN, 6–6
?, 6–1
?PASS, 9–1
?SHUTDOWN, 6–6
:$=, 11–5
:*=, 11–5
:=, 11–5
(F), 9–1
(FAIL), 9–1
(I), 9–1
(INCONC), 9–1
(P), 9–1
(PASS), 9–1
#DR0, 4–4
#DR1, 4–4
#WD0, 4–4
#WD1, 4–4
#WD2, 4–4
#WD3, 4–4
#WD4, 4–4
#WD5, 4–4
#WD6, 4–4
#WD7, 4–4
–", 7–2, 11–4
–>TM, 13–2

A

ALL_LINES, 3–5
ATCM, 6–1
AUTO_SET_PARAM, 11–4
AUTO_SET_SUITE, 7–4

B

BETWEEN?, 12–1

C

CASE;, 7–2
CASE:, 7–2
CASE_DISPLAY, 12–1
CASE_SEPARATOR, 4–10
CGS, 8–1
CGS_HEADER_DATE, 4–10
CHAR#REPORT, 4–5
CLIENT–STRING, 4–3
COMMAND, 3–1
COMPLETE#REPORT, 4–5
CONF_TEST, 5–1
CURR_CGS_SEL?, 8–3
CURRENT–CGS–PFA, 8–3

D

DEBUG, 12–1
DEBUGM_FALSE–>, 12–2
DEBUGS_FALSE–>, 12–2
DECIMAL_PARAM, 11–2
DECIMAL_PICS, 11–2
DEFAULT;, 6–5
DEFAULT:, 6–5
DEFAULT=, 6–5
DESELECT_CGS, 8–2
DESELECT_CGSP, 8–2
DISKLOG_FAIL, 4–4
DISKLOG_INCONC, 4–4
DISKLOG_PASS, 4–4
DOER_CASE1;, 12–6
DOER_CASE1:, 12–6
DOER_CASE2;, 12–6
DOER_CASE2:, 12–6
DOER_ERRORV, 12–7
DOER_FAIL, 12–6
DOER_GROUP1;, 12–6
DOER_GROUP1:, 12–6
DOER_GROUP2;, 12–6
DOER_GROUP2:, 12–6
DOER_INCONC, 12–7
DOER_PASS, 12–6

DOER_RUN_SEL, 12–7
DOER_STEP1;, 12–6
DOER_SUBGROUP1;, 12–6
DOER_SUBGROUP1:, 12–6
DOER_SUBGROUP2;, 12–6
DOER_SUBGROUP2:, 12–6
DOER_SUITE1;, 12–6
DOER_SUITE1:, 12–6
DOER_SUITE2;, 12–6
DOER_SUITE2:, 12–6
DSP_VERS, 12–6

E

END_LINE, 3–5
END_TEST_DOER, 12–7
ENUM_PARAM, 11–2
ENUM_PICS, 11–2
ERROR_VERDICT, 9–1
EXEC_TS, 13–3
EXIT–>, 6–6

F

F, 9–1
F–>D, 12–2
F1_PICS, 11–3
F2_PICS, 11–3
FAIL, 9–1
FAIL–IMMED–DISPLAY, 4–12
FALSE–>, 12–2
FORM#FEED, 4–10

G

GROUP;, 7–3
GROUP:, 7–3
GROUP_SEPARATOR, 4–10

H

HEADER–STRINGS, 4–3
HELP, 3–1
HEX#REPORT, 4–5
HEXSTRING_PICS, 11–3
HORIZONTAL–BAR, 4–10
HORIZONTAL=BAR, 4–10

I

IF?, 6–2
INC, 9–1
INCONC, 9–1
INCONC–IMMED–DISPLAY, 4–12
INDENTING, 12–5

L

L!, 6–1
L1_COMPLETE#REPORT, 4–6
L1_NONE#REPORT, 4–6
L1_SHORT#REPORT, 4–6
L2_CHAR#REPORT, 4–6
L2_COMPLETE#REPORT, 4–6
L2_HEX#REPORT, 4–6
L2_NONE#REPORT, 4–6
L2_SHORT#REPORT, 4–6
L3_CHAR#REPORT, 4–6
L3_HEX#REPORT, 4–6
L3_MNEMONIC#REPORT, 4–6
L3_NONE#REPORT, 4–6
L3COMPLETE#MODE, 4–6
L3MEDIUM1#MODE, 4–6
L3MEDIUM2#MODE, 4–6
L3SHORT#MODE, 4–6
LEAVE_PICS_SELECT, 12–7
LINE_AFTER_VERDICT, 4–10
LOAD, 3–1
LOAD_PARAM, 3–3
LOCK_SELECT, 8–2
LOCK_SELECTP, 8–2
LOCKED?, 8–3
LOG, 4–3
LOG#CASE, 4–3
LOG#LONG, 4–4, 4–5
LOG#OFF, 4–3
LOG#SHORT, 4–4, 4–5
LOG–DRIVE, 4–4

LOG_SELECT, 4–3
LOGHEAD–PARAMLIST–OUTPUT, 4–3

M

M–TMTM, 13–3
METHOD, 11–6

N

NO_LOOP!, 6–6

O

ONE_FILE_PER_CASE/GROUP, 4–4
ONLY–CASES–WITH–VERDICTPASS, 4–9

P

P, 9–1
P_ACCESS, 11–6
P_CHANGE, 11–6
P_DEBUG, 11–7
P_EXECUTE, 11–6
P_SCREEN, 11–6
PASS, 9–1
PASS–IMMED–DISPLAY, 4–12
PD, 9–1
PICSL;, 11–4
PICSL:, 11–4
PRINT, 3–5
PRINT#CHAR, 4–7
PRINT#COMP, 4–7
PRINT#HEX, 4–7
PRINT#MED, 4–7
PRINT#MED2, 4–7
PRINT#MED3, 4–7
PRINT#MED4, 4–7
PRINT#MED5, 4–7
PRINT#OFF, 4–7
PRINT#SHRT, 4–7
PRINT–L1–REPORT–TYPE, 4–6
PRINT–L1REPORT–TYPE, 4–8
PRINT–L2–REPORT–TYPE, 4–6
PRINT–L2REPORT–TYPE, 4–8
PRINT–L3–REPORT–MODE, 4–6
PRINT–L3–REPORT–TYPE, 4–6
PRINT–L3REPORT–TYPE, 4–7
PRINT–L4REPORT–TYPE, 4–7
PRINT–REPORT–TYPE, 4–5
PRINT–SCCPREP–TYPE, 4–7
PRINT–TCAPREP–TYPE, 4–7
PRINTLOG_FAIL, 4–5
PRINTLOG_INCONC, 4–5
PRINTLOG_PASS, 4–5

Q

QUIT_MENU, 3–1
QUIT_STAT, 3–6
QUIT_VLOG, 3–5

R

REGUILATORY_CAMPAIGN, 8–2
REGULATORY_CAMPAIGN, 4–9
REMAIN, 6–7
REMAIN_DEFAULT, 6–7
RESCAN, 6–6
RESTRICTED_CASES/GROUPS/SUITES, 4–9, 8–2
RETURN, 6–7
RUN–SELECTION–MODE, 4–9
RUN_SEL, 3–4, 9–2

S

SAVE, 3–1
SAVEPARAM, 3–3
SEL?, 8–3
SEL–BIT–TABLE, 8–2
SEL_ALL, 3–4
SEL_NONE, 3–4
SELECT+–, 3–4
SELECT_CGS, 8–2
SELECT_CGSP, 8–2
SELECTED?, 8–1

SET_PARAM, 11–4
SET_SUITE, 7–4
SHORT#REPORT, 4–5
SPY_OFF_TTCN, 12–2
SPY_ON_TTCN, 12–2
STA, 12–1
START_CASE_SELECT, 12–7
START_TEST_DOER, 12–7
STARTLINE, 3–5
STATIS, 3–4
STEP;, 7–2
STEP:, 7–2
STRING_PICS, 11–3
SUBGROUP;, 7–3
SUBGROUP:, 7–3
SUITE;, 7–3
SUITE:, 7–3
SUT–STRING, 4–3

T

T, 12–1
T", 12–1
TEST–LAB–STRING, 4–3
TM{, 13–2
TM=, 13–2
TMTM.B, 13–3
TMTM.B1, 13–3
TMTM.B2, 13–3
TMTM.B3, 13–3
TMTM.B4, 13–3
TMTM.B7, 13–3
TMTM.B8, 13–3
TREE, 12–5
TT_TMTM, 13–4
TTCN, 1–1
TTCN_SUITE_INIT, 5–2

U

ULTIMATE_FAIL, 6–6
UNLOCK_SELECT, 8–2
UNLOCK_SELECTP, 8–2

V

V., 12–3
V.", 12–3
V.C", 11–4, 12–3
V.CAPTURE, 12–4
V.DATE, 12–3
V.DUMP, 12–3
V.H, 12–3
V.HB, 12–3
V.HH, 12–3
V.ID., 12–3
V.PRINT, 12–4
V.R, 12–3
V.RAM, 12–4
V.RAMBUF, 12–4
V.SCREEN, 12–4
V.SCREEN+PRINT, 12–4
V.SELECTED, 12–4
V.SPACES, 12–3
V.TIME, 12–3
V.TRACE, 12–4
V.TRACE+PRINT, 12–4
V.TYPE, 12–3
V.TYPH, 12–3
VCR, 12–3
VERDICT?, 9–2
VERDICT–LOG, 4–11
VERDICT–LOG–FAIL–IMMED, 4–11
VERDICT–LOG–INCONC–IMMED, 4–11
VERDICT–LOG–PASS–IMMED, 4–11
VERDICT–STATISTICS–GROUP, 4–11
VERDICT–STATISTICS–SUBGROUP, 4–11
VERDICT–STATISTICS–SUITE, 4–11
VERSION, 3–1
VLOG, 3–1, 3–5

W

WAIT_TIME_AFTER_CASE, 4–9
WTD, 12–5

Y

YES_LOOP!, 6–6